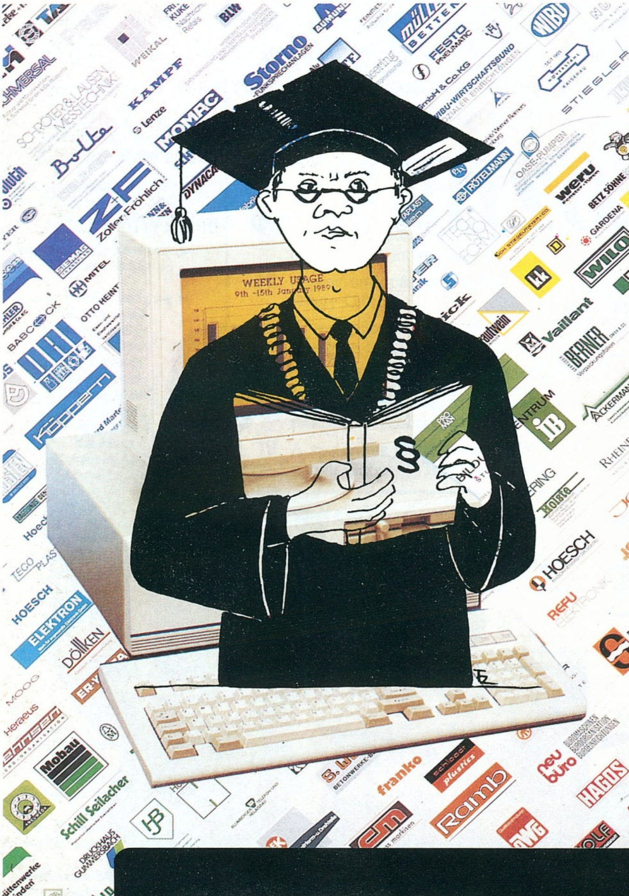
1991 / SZEPTEMBER

ÁRA: 196 FT

ALAPLAP

MIKROSZÁMÍTÓGÉP MAGAZIN MÁGNESLEMEZ MELLÉKLETTEL



PC Tools 7.0 — a feneketlen láda

Veszélytelen lövöldözés

Bill Gates a rendszerek jövőjéről

A MÁGNESLEMEZEN:

A COM-gyűjtő láda
Könyvtártörlő segédprogram
GEM operációs rendszer XIV.
Modula évszakok
Céllövölde

ARJ — a tömörítők élén

A PC szereptévesztése

Assembly: kéj és kín egyszerre

„Egyéb javaslatai, észrevételei...”

A HÓNAP TÉMÁJA:

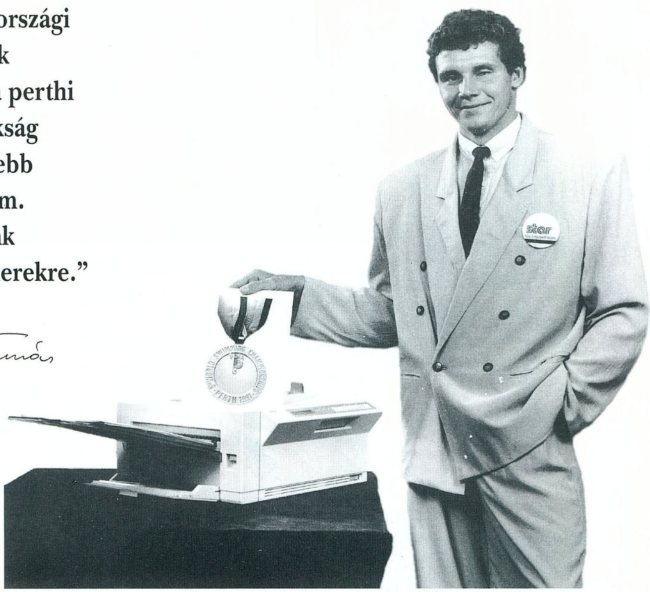
COPYRIGHT *.*

Sztárok... Ha találkoznak...

● STAR. Aki járatos a számítástechnikában, szerte a világon ugyanarra a kiváló minőségű, japán gyártmányú nyomtató családra gondol e szó hallatán. Ez a név ma már összefonódott a minőség, a szép formák, a nagy választék és a szolid árak fogalmával. ● STAR. Kínálatával és szolgáltatásaival a világszínvonalat jelenti Magyarországon. Saját szervízhálózata és szerteágazó viszonteladói rendszere biztosítja ezt. ● SZTÁR. Így nevezzük azokat, akik saját területükön egyedülállót nyújtanak. S hogy ez a kétféle sztár fogalom hogyan kapcsolódik össze? Azok a világhírű sportolók, akiket a Star segített sikerük elérésében, tudják erre a választ.

„A Star magyarországi disztributorának támogatásával a perthi úszóvilágbajnokság legeredményesebb sportolója lettem. Együtt készülünk a barcelonai sikerekre."

Darnyi Tamás



Ogilvy&Mather

star

Exclusive distributor: HRP Consultants S.A.R.L. Jersey
Képviselet és bemutatóterem: 1051 Budapest, Nádor utca 32.
Tel.: 132-1811, 132-7534 Telefax: 131-8177

INFORMÁCIÓKÉRÉS: 09 ▲

ALAPLAP

Mikroszámítógép magazin
mágneslemez melléklettel
Megjelenik havonta

Főszerkesztő:
Faklen Pál

Főszerkesztő-helyettes:
Varga János

Szerkesztő:
Jakab Ágnes

Szerkesztőségi titkár:
Sziebig Andrea

A mágneslemez melléklet
és a Közkincs szerkesztője:
Verebély Pálné

A Lemezkalauz szerkesztője:
Vékony Tamás

A szerkesztőbizottság tagjai:
Barna László
Boros György
Broczkó Péter
Brüll Károly
Farkas Ernő
Herczeg József
Horváth Imre
Kassay Árpád
Kis János
Kovács P. Attila
Kónya László
Nagy Gábor
Pintér Gábor
Zoltai Péter

Szerkesztőség, kiadó és
hirdetésszervezés:
XI., Karolina út 17.
Budapest 1251
Telefon: 185-1584
Fax: 185-2221

Felelős kiadó:
Sebestyén Ilona igazgató
Cédrus Informatikai Rt.

Nyomdai előkészítés:
Tipoprint Kft., Budapest
Nyomtatás:
Zalai Nyomda, Zalaegerszeg
Felelős vezető: Galla József

Terjeszti a Magyar Posta.
Előfizethető a hírlapkézbesítő
postahivataloknál és a Posta
Hírlapelőfizetési és Lapellátási
Irodájánál (XIII., Lehel u. 10/a,
Budapest 1900), vagy átutalással
a 215-96162 pénzforgalmi számra.
Példányonkénti ár: 196 Ft
Évi előfizetési díj: 2 352 Ft

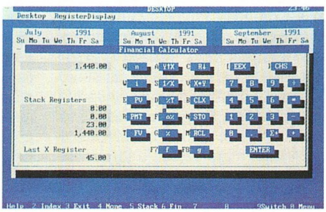
Külföldre terjeszti a Kultúra,
Pf. 149, Budapest 1389

HU ISSN 0865-9788

A HÓNAP TÉMÁJA: COPYRIGHT *.*

TÉMABŐVÍTŐ

SZERSZÁMOSLÁDA



KÖZKINCS

SOLARSOFT LEMEZKALAUZ

GÉPRAJZ

ADATRENDEZŐ

KILÁTÓ

ABAKUSZ

FOGÓDZÓ

PROGRAMOZÁSTECHNIKA

VISSZACSATOLÁS

PALETTA

MÁGNESLEMEZ MELLÉKLET ❑

Könyvtártörlő segédprogram
A GEM operációs rendszer XIV.
A COM-gyűjtő láda
Modula évszakok

A címlapon és a cikkek között Feleki Zoltán karikatúrái



Védett szellemi termék

Kit véd a copyright?

Kit védene? A szerzőt! — feleljük, szinte gondolkodás nélkül. Ám a dolog a valóságban korántsem ennyire magától értetődő, és a kérdés megéri, hogy elgondolkodjunk rajta, mielőtt kimondjuk a látszólag kézenfekvő választ.

A könyvnyomtatás feltalálása előtt a szerzői jog kérdése fel sem merült. Kr.u. 533-ban Justinianus nyugatrómai császár híres törvénykönyveiben (Corpus iuris) a hagyományra és Gaius római jogtudós korábbi döntésére hivatkozva még azt állapította meg, hogy valamely pergamenre vagy papiruszra írt vers vagy történet annak a tulajdonába kerül, akié az anyag, amire írták, még ha színaranyból is készültek volna a betűk.

Az európai kontinensen, nem sokkal azután, hogy Gutenberg találmánya viharos gyorsasággal terjedni kezdett, előtérbe került a jogvédelem problémája — de nem a szerzők, hanem a nyomdászok érdekében. Luther 1525-ben „Eyn Vermanung an die Drükker" című felhívásában az illetéktelen utánnyomások ellen háborgott, útonálló rablókhoz és tolvajokhoz hasonlítva az egymás romlását előidéző reprodukciós kiadókat. Baselban 1531-ben megtiltották, hogy valamely kiadványt három éven belül bárki is utánnyomjon. 1640-ben Johann Schopius kért császári védelmi jogot „privilegium impressorium" iránt — a monopólium örököseire is kiterjedt. 1761-ben a francia királyi tanács La Fontaine unokáinak adott privilégiumot nagyapjuk műveire. 1777-ben XVI. Lajos elrendelte, hogy a szerzőnek adott monopólium annak élete végéig szól, és ha nem adta el másnak, akkor örököseire is átszáll.

Angliában 1580-ban jelent meg először a „right of copy" kifejezés a Stationer's Company kiadványain. 1649-ben Milton követelte, hogy „a művén fennálló tulajdonjogát minden szerzőnek biztosítani kellene, mind életében, mind halála után." 1709-ben Anna királynő elrendelte, hogy a művet kiadó nyomdát 14 évre szóló utánnyomási tilalom védje, majd ezután a jog — ha a szerző életben van — újabb 14 évre a szerzőre visszaszáll, azonban ha meghalt, az utánnyomás joga közkinccsé válik. 1769-ben a King's Bench úgy döntött, hogy a szerző irodalmi tulajdonjoga éppúgy korlátlan időtartamú, mint a dolgokon fennálló tulajdonjog. 1774-ben a Lordok Háza a Donaldson v. Beckett ügyben kimondotta, hogy a korlátlan idejű védelmet a mű első kiadásakor fölváltja az időben korlátozott védelem. Ezzel a döntéssel született meg a modern szerzői jog.

Milyen jellegű jog tulajdonképpen a szerzői jog? Joseph Koller (1880) szerint a megfogható, anyagi javakhoz hasonlóan léteznek bizonyos anyagtalan javak, értékek; ez utóbbiakra a dologi jog szabályait kell kiterjeszteni. Otto von Gierke (1895) véleménye már közelebb áll a mai felfogáshoz. Szerinte a szerzői jog elsődlegesen a szerző személyiségéhez tapad — akárcsak a becsület védelme vagy a jó hírnévhez való jog —, és csak másodlagos, járulékos célja a vagyoni érdekek megóvása. Alexander Elster (1920) szerint viszont a szerzői jog egyszerűen az ipari és kereskedelmi versenyjog része, a plágium vagy a jogbitorlás nem közvetlenül a szerző személyiségét vagy vagyonát sérti, hanem a társadalomnak a tisztességes piaci versenyhez fűződő érdekeit.

Az első magyar szerzői jogi törvény, az 1884. évi XVI. tc. tervezetét Arany László — a nagy költő fia, maga is íróember — készítette elő. A jogszabály az anyagtalan javak tulajdonjogi koncepcióján alapult.

S hogy hol tartunk ma, illetve mennyiben érinti ez a számítógépek környezetében tevékenykedők életét, zömmel erről szólnak e számunkban a hónap témájának írásai, részben a tárgykör hazai művelőinek tollából, részben számítástechnikai cégek és szakértők nézőpontjából.

Pálfordulás

Szabadalom és szerzői jog

A számítógépek a második világháborút követő években törtek ki a kizárólagos katonai alkalmazással együttjáró szigorú titoktartás zárt falai mögül — ma már tudjuk, hogy a háború alatt a német haditengerészet rejtjelezési rendszerét az Enigma nevű komputer segítségével fejtették meg. Mindenesetre az a tény, hogy az elektronikus számítógépek (először persze a nagygépek) betörtek a polgári piacra és egyúttal elterjedtek a magas szintű programozási nyelvek, fölvetette a szoftverek jogvédelmének a problémáját.

Különösen fölpörgött a jogvédelem kérdése akkor, amikor a miniatürizálás eredményeként megszülettek a mini- és mikrogépek. Ezzel olyan feladatok egész sora vált kisvállalkozás keretében, sőt otthoni íróasztal mellett, könnyűszerrel, matematikai felkészültség vagy számítástechnikai szakértelem nélkül is kényelmesen megoldhatóvá, amire néhány évvel korábban még gondolni sem lehetett.

A 60-as években és a 70-es évek elején heves viták dúltak a szoftverek szabadalmaztathatósága körül. Számítógépes szakemberek és jogászok százainak tollából, persze elsősorban az Egyesült Államokban és Nyugat-Európában, könyvtárnyi szakirodalom született. Záporoztak az érvek és az ellenérvek.

A szabadalmi védelem hívei kifejtették, hogy a szoftver alapjában véve ipari jellegű termék, és felhasználása is túlnyomó részben a termelés területén, az üzleti világban történik. Indokolt tehát, hogy iparjogvédelmi eszközökkel biztosítsák a hozzá fűződő jogosultságokat és érdekeket. Az ellenfelek többek között azzal érveltek, hogy a program nem valóságos anyagi termék, csupán logikai konstrukció; a szabadalmi védelem pedig csak a természeti erőket közvetlenül felhasználó eljárásokra, azokat megtestesítő termékekre alkalmazható. Ugyan kérem! — vágtak vissza a vitapartnerek — hiszen végső soron a szabadalmi jog körébe tartozó összes ipari gyártási eljárás logikai-szellemi konstrukció, és ha jogilag szabadalmi védelem adható a mesterségesen kitenyésztett új állat- és növényfajtákra, miért ne lehetne ugyanezt biztosítani a szoftverekre is?



Amíg a jogtudósok okoskodtak és a szellemi csaták megtöltötték a szakfolyóiratok hasábjait, a jogalkalmazó hatóságok, a szabadalmi hivatalok és a bíróságok — ha nem is teljes következetességgel, de az esetek túlnyomó többségében — tetszetős vagy kevésbé tetszetős jogi érveléssel elutasították a szoftverek szabadalmaztatása iránti kérelmeket.

Úgy vélem — anélkül, hogy kétségbe vonnám a tekintélyes tudorok és testületek jóhiszeműségét és szakértelmét —, hogy e döntések mögött korántsem csupán a döntéshozók tiszta, veretes jogelméleti meggyőződése rejlett. Ugyanis az 1960-as évek végéig a legnagyobb gazdasági hatalommal rendelkező számítógépgyártók, elsősorban az IBM, ingyen adták a gépet megvásárló ügyfeleknek a szoftvert. Ennek költségét — természetesen tisztes profittal együtt — beszámították a hardver eladási árába. Érdekük volt tehát, hogy ingyen vagy olcsón jussanak hozzá mások szellemi munkájának eredményeihez, ezért ebben az időszakban nem támogatták, sőt élesen ellenezték a szoftverek szabadalmaztathatóságát.

1969-ben a trösztellenes törvény rákényszerítette az IBM-et, hogy a szoftvereknek külön árat állapítson meg. Ekkor viszont elsőként nyújtott be javaslatot a jogvédelemre, de miután mindaddig élesen ellenezte a szabadalmi védelem megadását, most már a szerzői jogi koncepció mellett kényszerült kiállni — amely egyébként sokkal erősebb védelmet nyújt a szabadalmi iparjogvédelemnél. A helyzet megváltozásával a jogtudomány művelői alkalmatlannak találták ugyan az iparjogvédelmet biztosító szabadalmi jogot alkalmazni a szoftverekre, de egyben megállapították, hogy az addig kizárólag a művészeti és tudományos alkotásokra alkalmazott szerzői jog kiválóan alkalmas a számítógépi szoftverek védelmére.

Manapság viszont egyre inkább úgy tűnik, hogy a nagy vállalkozások mind kevésbé élnek a formális jogvédelem eszközeivel. A piacon elfoglalt előnyös helyzetük, a reklám és a terjesztés eszközeinek kézbekaparintása lehetővé tette, hogy egy-egy szoftverrel egyidejűleg árasszák el az egész piacot, és rendkívül rövid idő alatt söpörjék be a termékkel elérhető nyereséget. A haszonból azért egy-két százalékot juttatnak az alkotóknak is, általában terjesztési szerződés alapján.

Legyen tehát bármilyen zseniális a szoftver alkotója, nagyszerű a programja, a nagy terjesztőkkel történő „kooperáció" nélkül nem rúghat igazán labdába. A kitűnő francia filozófus, Michel Foucault 1969-ben írta le — igaz, egészen más kontextusban — a társadalomtudományokban azóta szállóigévé vált, paradoxnak látszó kérdést: „Mi a szerző? Mit számít, ki beszél?" Valószínű, hogy a mai, de méginkább a holnapi szoftver alkotója elmondhatja: „Ki a szerző? Mit számít? Az számít, hogy kinek van pénze a terjesztésre!"

Kálmán János

A szoftver jogi védelme

A program — ipar vagy művészet?

A számítógépi programok védelmével alig több mint negyed évszázada foglalkoznak. Először alapvető kérdésekre kellett választ keresni. Mi a számítógépi program, a szoftver jogi minősítése? Hol helyezkedik el a jog területén? Ha most visszatekintünk arra, hogyan zajlott le a szoftver jogterületi honosítása, akkor fel kell idéznünk, hogy a szerzői jognak a szoftverekre való alkalmazásával eleinte nem nagyon szimpatizált senki.

Maga a kérdés az iparjogvédelem terén vetődött fel, mert a szoftver eredendően gépekkel, berendezésekkel (hardverrel) kapcsolatos, és mert ipari és kereskedelmi vállalatok működésével függ össze. A szakma inkább a maga számára legkézenfekvőbb eszközhöz, a szabadalomhoz igyekezett nyúlni, és nem kívánt beletörődni abba, hogy a szoftvert, pontosabban magát a programot az arra illetékes hivatalok és bíróságok világszerte egyre kevésbé ismerték el szabadalmaztatható találmányként.

Ez a helyzet körülbelül a hetvenes évek végéig tartott. Addigra a sok elutasító bírói ítélet után már különböző nemzeti törvények is kimondták, hogy komputerprogramok általában nem szabadalmazhatók. A francia törvény például már 1968-ban leszögezte ezt. Hogy közép-kelet-európai országot is említsek, kimondta ugyanezt a lengyel törvény is, 1972-ben. Az USA-ban a Legfelső Bíróság 1972-ben hozott ilyen értelmű döntést a Gottschalk v. Benson ügyben, amelyet az Amerikai Szabadalmi Hivatal kiterjesztően alkalmaz. Amikor az 1973. évi Európai Szabadalmi Egyezmény Münchenben létrejött, külön szakasza állapította meg, hogy az adatfeldolgozó berendezésekhez készült programok nem tekinthetők találmánynak. Ez volt az első nagy kiábrándulás.

A honosítás után

Körül kellett nézni a jog egyéb területein; megindult a programvédelem kibontakozásának második fázisa. Két választás volt; az egyik: csinálunk egyfajta sui generis, önálló védelmet, számolva annak minden problémájával. Csak egy ilyet említenék: az idő gyorsan múlt, a szoftveralkotás már burjánzott és vele együtt elharapódzott a szoftverlopás; a hardverek üzemeltetése, az elektronika fejlődése nélkül már nem volt elképzelhető ipari-kereskedelmi fejlődés, a programvédelem rendező eleme pedig hiányzott. Azon gondolkodni, hogyan képezzünk valami sui generis védelmet, többünknek rendkívül kalandos vállalkozásnak tűnt.

A másik rendelkezésre álló megoldást a szerzői jog kínálta: nem minősülhet-e esetleg a program szerzői alkotásnak? Világviszonylatban elkezdődött egy olyan folyamat, amelyben Magyarország az elsők között vett részt. Az Egyesült Államokban először 1964-ben regisztrált számítógépi programot az ottani Copyright Office, az USA Szerzői Jogvédő Hivatala, amely a Kongresszusnak szakosított szervezete.

1968 óta

Magyarországon az irodalomban először 1968-ban vetettem fel ezt a kérdést. 1972-ben pedig már a Fővárosi Bíróság hozott olyan ítéletet, amelyben egy olajkutatási módszert program formájában rögzítő alkotásnak szerzői jogi védelmét állapította meg, méghozzá az alkotófolyamat lépcsőzetes, összetett jellegére tekintettel, több fázisára nézve.

Ez azért is jelentős lépés volt, mert egyben bizonyította, hogy a szerzői jogi törvény a csak 1983-ban meghozott explicit végrehajtási szabályok nélkül is alkalmazható, úgy ahogy van — minden újszerű szerzői jogi alkotás védelmére, ami egyébként eredeti műnek minősül. Később ez a döntés azáltal is megszilárdult, hogy a törvényt végrehajtó 9/1969. évi művelődési miniszteri rendelet 1983. évi módosításával a szerzői alkotások példálódzó felsorolásában kifejezetten megemlítették a programokat is, mint a szerzői jogi védelem tárgyát, majd néhány speciális végrehajtási szabályt is hoztak a szerzői jogról szóló 1969. évi III. törvény alkalmazásáról komputerprogramokra.

A sui generis védelem kérdését illetően, az eddig jelzettek fényében, a következő tanulság adódik: nem új terület kellett ahhoz, hogy honosítani tudjuk a szoftver védelmét. A szerzői jog birodalmában, a szellemi alkotások nagy glóbuszán igenis ott a helye a számítógépi programoknak is. A védelem alkalmazása azonban valóban sui generis szabályokat igényel. (A szerzői jog fejlődése során korábban is történt hasonló. A film védelmére nem lehetett a könyvkiadás szabályait alkalmazni.)

A mai gondok

Itt érkezünk el problémánk fejlődésének jelenlegi fázisához. Amikor a területi honosítás a szerzői jog birodalmában megtörtént, s amikor tudjuk, hogy a szerzői jog alkalmazása ezen a területen „sui generis végrehajtást" igényel, akkor már kézenfekvő: ma a szerzői jog adekvát alkalmazásának kérdéseivel kell foglalkoznunk.

Ezek a kérdések ismét többfélék — probléma lehet, hogy ki a szoftver szerzője, mikor áll fenn szerzőtársi viszony vagy társszerzőség; mikor eredeti egy szoftver, milyen felhasználási szerződéseket kell kötnünk. A szerzői jogon belül e vonatkozásokban speciális körülményekkel kell szembenéznünk. Tovább is lépnék azonban: szomszédos jogterületekre is áthatott ez a fejlődés, mert egyszerre egy csomó munkajogi rendezetlenség vált zavaróvá — hiszen a munkaviszonyban alkotott számítógépi program szerzői jogi elbírálása ismét sajátos szabályokat igényel. Ezzel kapcsolatban szakértői eljárásokat, fellebbezési eljárásokat, a kollektív szerződés vagy munkaügyi szabályzatok átalakí-

tását, külön szoftverszabályzat létesítését stb. kell rendezni. Még tovább nézve: az egész pénzügyi mechanizmust érinti, hogy minek a terhére számoljuk el a szoftverszerzőknek járó díjakat, ezek milyen adózás alá essenek.

A tárgykör megközelítési irányai

Elérkezve idáig, kíséreljük meg röviden áttekinteni, hogy a szoftver szerzői jogvédelme a szellemi alkotások összterületére kihatóan milyen aktuális felismerésekkel jár, mire kell odafigyelnünk. Abból indulnék ki, hogy az egész zsákutcába jutás, ami a 60-as évek végét és a 70-es évek elejét jellemezte, amikor iparjogvédelmi területen kerestük a megoldást a komputerprogramok védelmére, elsősorban mire volt visszavezethető. A jogászok tudatában ui. nem élt szabatosan, mi tartozik az iparjogvédelem és mi a szerzői jogvédelem körébe.

A szerzői jog védi az irodalom és a művészet alkotásait, itt írók, zeneszerzők, festők kereshetnek védelmet: ez a hagyományos kép. Az iparjogvédelem pedig védi mindazt, ami az ipari és a kereskedelmi érdekeket szellemi értékelemekkel mozdítja elő. Az egyik tehát egy műfajorientált, durva megközelítés volt, a másik pedig kereskedelmi, profitközéppontú, érdekvédelmi tájolás. Ilyen alapon vetődhetett fel, hogy miután a komputerprogram hagyományos értelemben se nem irodalom, se nem művészet, ugyanakkor az ipar érdekeit elősegítő szellemi érték, induljunk el az iparjogvédelem irányában.

Világszerte nagy szerepe volt a számítógépi program szerzői jogi védelme elismerésének abban, hogy a szerzői jog tárgyának — a szabadalmazható találmány fogalmától szabatosan elhatárolva — általános műfogalomkénti meghatározásához sikerült eljutni. Az iparjogvédelemnek kezdettől fogva volt bizonyos definíciókényszere: mivel többféle védelmet egyesített saját területén, definiálnia kellett azok tárgyát. Hiszen az iparjogvédelem nemcsak szabadalom adásából, találmányok védelméből áll, hanem oda sorol a védjegyoltalom, a mintaoltalom, sőt a tisztességtelen gazdálkodás tilalma is, a legáltalánosabb értelemben.

Ezeket a területeket az iparjogvédelem körén belül egymástól el kellett választani. Ezért az iparjogvédelem terén precíz, szabatos definiálásra került sor, mindenekelőtt a szabadalmazható találmány vonatkozásában. Mindez sokáig nem történt meg a szerzői jogi alkotások tekintetében. Így, amikor a szerzői jogi mű fogalmának általános elemeit kezdtük keresni, ezt egy kicsit a találmány védelmével szembesülve tettük. Ezzel összefüggésben három kérdésre kell válaszolnunk.

Mit, mi ellen, milyen eszközökkel?

Szabadalmazható találmány az olyan új, haladást jelentő és műszaki megoldás, amely a gyakorlatban alkalmazható. A találmányok szabadalmi oltalmáról szóló 1969. évi II. tv. a szabadalmazható találmány általános definicióját öt szakaszában részleteiben is értelmezi.

De mit jelent az, hogy új? Ami nem annyira közismert, hogy szakember megvalósíthassa. Mit jelent az, hogy haladást jelentő? Hogy újabb szükségletet elégít ki, vagy az eddiginél jobban elégít ki valamely már kielégített szükségletet. Mit jelent az, hogy műszaki? Ami termékben vagy termelési eljárásban változást eredményez. A gyakorlati alkalmazhatóság pedig azt jelenti, hogy a találmány azonos eredménnyel ismételten megvalósítható.

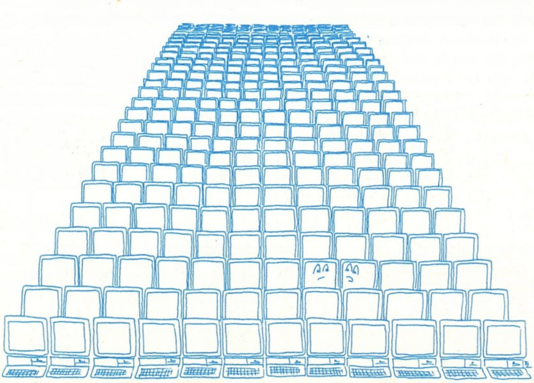
Ezzel kapcsolatban megjegyzem, hogy nem azért nem lehetett szabadalmaztatni a számítógépi programokat, mert a szabadalmi jog nem fejlődőképes. A szabadalmi jog a maga keretein belül állandóan gazdagodik; ezt éppen a műszaki követelmény tág értelmezésénél láthatjuk — nem hinném, hogy mondjuk egy vörös galambfajta kitenyésztése műszaki probléma lenne (ez inkább biológiai kérdés), mégis szabadalmazható. A mi törvényünk végrehajtási rendelete már 1969-ben kifejezetten leszögezte, hogy műszaki jellegűnek kell tekinteni a növénytermesztési és állattenyésztési eljárásokat is; műszakinak tekintendő tehát végső soron a természeti törvények alkalmazása.

A találmány kritériumai

Lényegében tehát mit védünk találmányként? Valaminő, meghatározott jellegű megoldást. A hagyományos szerzői jogi megközelítés — azon túl, hogy műfajorientált volt — ezzel kapcsolatban egyszerűen úgy vélte, hogy a tartalmat védi a szabadalom, a szerzői jog meg a formát. A formavédelem ilyen didaktikus, banális megszövegezésben nagyon sokszor — oktatás keretében is — elhangzott, és ez generációkat vezetett félre. Gondoljunk azonban arra, hogy ha egy regényt megfilmesítenek, azt adaptálják, az ilyen felhasználással szemben már nem az eredeti formát védi a jog; talán egy szó sem hangzik el a regény szövegéből a film dialógusaiban; és ha lefordítanak egy könyvet idegen nyelvre, semmi sem marad eredeti kifejezési formájából, mégis védjük a jogosulatlan fordítással szemben.

Végső soron tehát a szerzői joggal is a tartalmat védjük. A találmány meghatározásával szemben a szerzői jog absztrakt műfogalomhoz jut el — amely szerint véd minden, azonosítható módon kifejtett, eredeti gondolatszövedéket.

Az azonosítható módú kifejtés mint kritérium két fontos sajátosságra utal. Az egyik: az alkotásnak nem kell rögzítettnek lennie (mert szóban is ki lehet azonosítható módon fejteni bizonyos műveket, vagy empirikusan elsajátíthatóan be lehet tanítani egy balettet, s improvizálni is lehet egy zeneművet).



– Gondoltad volna, hogy ennyi távol-keleti rokonunk van?

A másik, ezzel kapcsolatos felismerés pedig az, hogy a mű formája nem a védelem tárgya, hanem a védelmet lehetővé tevő követelmény — akkor is, ha nem kívánjuk meg rögzítését. A mű azonosítható módon történő kifejtése követelményének az a célja, hogy felismerhetővé tegye tartalmát és megkülönböztethetővé tegye azt.

A műnek eredetinek kell lennie, tehát nem utánzásnak; teljes vagy részleges plagizálás nem eredeti mű. Viszont: a műnek nem kell abszolút értelemben újnak lennie, mint ahogy azt találmányok esetén elvárjuk. Fontos továbbá, hogy a mű lényege valaminő, legtágabb értelemben vett gondolatszövedék, tehát nem egyszerűen ötlet, módszer vagy megoldás.

Egy ilyenfajta elhatárolás mellett világos, hogy egy eredeti komputerprogram kimeríti a szerzői jogi mű fogalmát, az alá szubszumálható; ezzel szemben önmagában nem jelent sem termékben, sem termelési eljárásban változást. Tehát logikus, hogy szabadalmazását általában meg kell tagadni.

A „gyakorlati alkalmazhatóság"

Következő kérdésünk arra irányul, hogy mi ellen védjük a szerzői művet, illetve a szabadalmat. Az utóbbit illetően már elhangzott a válasz: a szabadalmat a megvalósítással szemben védjük. A törvény szakaszai több ízben utalnak az ismételt megvalósíthatóság követelményére. Ezt a „gyakorlati alkalmazhatóság" értelmezése végett teszik. Az „alkalmazás" szó ugyanis korántsem egyértelmű. Mert ha számítógépi programra gondolunk, azt is alkalmazzuk akkor, ha valamely hardverben futtatjuk, ha egy probléma megoldása érdekében igénybe vesszük. Ám nem „valósítjuk" meg.

A megvalósítás azt jelenti, hogy azt az új műszaki megoldást, ami a szabadalomnak a tárgya, az életben azonos eredménnyel, ismételhetően realizálják. Nos, ez a fogalom a szerzői jogban irreleváns.

A szerzői jog nem a megvalósítással, hanem a megjelenítéssel, annak különféle formáival szemben véd, így: többszörözés, előadás, sugárzás vagy egy már sugárzott műnek kábeles továbbközvetítése, elsődleges vagy másodlagos, de mindenképpen megjelenítésével szemben. Láttuk tehát, hogy mit véd, és azt is láttuk, hogy mivel szemben véd a szerzői jog. A számítógépi program mindkét megközelítés felől nézve a szerzői jog körébe tartozik.

A tisztességtelen verseny ellen

Végül lássuk a harmadik kérdést: milyen eszközökkel védjük a művet, illetőleg az iparjogvédelem tárgyát.

Az iparjogvédelem legfontosabb eszköze a konstitutív, regisztrálással keletkező, eleve tanúsítható jog, amely a jogosultnak nyugodt, biztonságos érzést ad: kérem, itt a papír, ezt senki sem vitathatja, nem kell bíróság előtt bizonyítanom a jogomat. Okmány igazolja, hogy én vagyok a feltaláló, regisztrálják a védjegyemet, van ipari mintám stb. De ugyanakkor — és itt a tisztességtelen gazdálkodás tilalmára gondolok — van az iparjogvédelemnek másfajta eszköze is, amely bizonyos magatartástípusokat tilt.

Ezeknek a magatartástípusoknak és tilalmuknak nem felel meg minden esetben alanyi jog. Nem szükségképpen kizárólagos jog az, ami engem tisztességtelen verseny elleni tiltakozásra feljogosít.

Az iparjogvédelem eszköztárából a számítógépi programok esetében a regisztrálással konstitutíve keletkező, a jog meglétét eleve tanúsító kizárólagos jog intézménye nem vehető át. Igénybe vehető azonban az a másik általános fajtájú tilalom (a tisztességtelen verseny tilalmának nevezett korlátozó rendelkezés), ami meghatározott jellegű felhasználások esetén, de csak bizonyos körülmények fennforgásakor ad lehetőséget arra, hogy valamely számítógépi program jogosulatlan felhasználásának eltiltását követeljük.

Előnyben az előnyök

A szerzői jog ezzel szemben olyan kizárólagos jogot biztosít a komputerprogramon, ami nem regisztrálással keletkezik, és nem eleve tanúsítható. Mert a szerzői jognak az a jellemzője, hogy önmagában az alkotás tényénél fogva ismeri el — az alkotó szerző javára, alkotása tárgyára vonatkozóan — a kizárólagos jogot. Ez semminemű formalitástól nem függ, rögtön keletkezik, azonnal rendelkezésre áll; ez az előnye. Létezése azonban nem bizonyítható eredendően, mint az iparjogvédelmi regisztrálás esetében; ez a hátránya.

Az a fontos azonban, hogy ma, amikor egy számítógépi programot a megalkotása után, sőt sokszor már a készülése során is használatba vesznek a védelem azonnal rendelkezésre álljon és kizárólagos jog formájában legyen jelen. Ne függjön bizonyos körülményektől, mint pl. a tisztességtelen verseny tilalmával összefüggő védelem, amit most az 1990. évi LXXXIV. törvény szabályoz nálunk.

Megérzésem szerint — és a gyakorlat tükrében is — a programok felhasználásával kapcsolatban több előny adódik a szerzői jogi védelem jellegéből, mint hátrány. Nem beszélve arról, hogy a tisztességtelen gazdálkodás, illetve verseny elleni tilalmon kívül más iparjogvédelmi eszköz általában és eredményesen nemigen vehető igénybe. Aránylag ritkán van szó olyan programokról, amelyek valamely komputer funkcionális, elválaszthatatlan elemeként mint megoldás szabadalmazhatók. Más esetekben olykor lehet ugyan szó üzleti titok, hivatali titok védelméről, ezek azonban már nem szorosan vett iparjogvédelmi tényállások, és védelmük jóval kevesebbet ér bármely abszolút engedélyezési jognál.

A szerzői jogi védelem sikere ezek után az alkalmazásán áll — megfelelő szabályok útján.

Boytha György



A szoftver a külföldi szerzői jogban

A gazdagabbak erősebbek?

Az egyéni eredeti számítógépi programokat több tucat ország — közte az összes fejlett ipari állam — joga szerzői alkotásként (is) védi. Jellemző példaként a világ legnagyobb szoftvergyártó államainak „szoftver szerzői jogát" vázoljuk fel.

Németország

A német szerzői jog az 1985. évi módosítás nyomán a számítógépi programokat irodalmi műként rendeli védeni. További önálló jogszabályi rendelkezések hiányában — a magáncélú másolás esetét kivéve — a számítógépi programok szerzői jogi védelmére is a szerzői művekre általánosan irányadó jogszabályok vonatkoznak.

A bírói gyakorlat ugyanakkor néhány kérdésben mégiscsak sajátos álláspontot alakított ki. A Legfelső Bíróság a precedensként emlegetett „Inkasso-program" ítéletben a számítógépi programok vonatkozásában egy, a szerzői művekkel szemben támasztott eredetiség követelményét meghaladó, magasabb mércét alkalmazott. A nevezetes ítélet szerint ugyanis számítógépi programok esetében a megállapított egyedi vonásokat és az ezekből adódó összbenyomást egy átlagos szerző művével is össze kell vetni, és „csak az olyan alkotás, amely az átlagos tudású programalkotó teljesítményét lényegesen meghaladja, tarthat igényt a szerzői jogi védelemre". Az „Inkasso-program" ítélettel és következményeivel a német jogirodalomban néhányan egyetértenek, de az ott kifejtett elv merev alkalmazását a többség veszélyesnek tartaná.

Franciaország

A francia szerzői jog egy 1985-ben született törvényben rendelkezett többek között a számítógépes programok védelméről is. A jogszabály értelmében a szerző — ellenkező szerződési kikötés hiányában — köteles tűrni, hogy a felhasználó a megengedett terjedelemben a programot átdolgozza, nem élhet továbbá a kijavításhoz, illetőleg a visszavonáshoz való jogával sem.

A számítógépi programok tekintetében alkalmazandó 25 éves védelmi idő kivételesnek mondható a szerzői művekre általában megállapított, a Berni Uniós Egyezmény rendelkezéseihez igazodó védelmi időhöz képest (amelynek értelmében a mű a szerző életében és a halálát követő 50 éven belül védett). Ez nyilvánvalóan az említett egyezménybe ütközik, amivel szemben a rendelkezés védelmezői kevés meggyőző erővel hivatkoznak arra, hogy a szoftver „alkalmazott művészeti alkotás" lenne. Utóbbira ugyanis az egyezmény rövidebb védelmi időt is enged. Sajátos rendelkezéseket állapít meg továbbá a jogszabály a számítógépi programok szerzői jogának megsértésével kapcsolatos eljárásra is.

Amerikai Egyesült Államok

Az Egyesült Államok szerzői jogi törvényének 1980. évi módosításában szereplő meghatározás értelmében: a számítógépi program olyan megállapítások vagy utasítások sorozata, amelyet meghatározott eredmény elérése céjából, közvetlenül vagy közvetett módon használnak fel a számítógépben.

Nem tekinthető a szerzői jog megsértésének, ha a számítógépi programot annak jogos tulajdonosa a számítógépi felhasználás vagy biztonsági másolat céjából lemásolja vagy átdolgozza.

Az ítélkezési gyakorlat során kialakított állásfoglalás szerint a szerzői jogi védelem nemcsak a számítógépi program formai, betű szerinti elemeire, de annak szerkezetére és felépítésére is kiterjed, és nincsen különbség a védelem szempontjából a forrásprogram és a tárgyi program között.

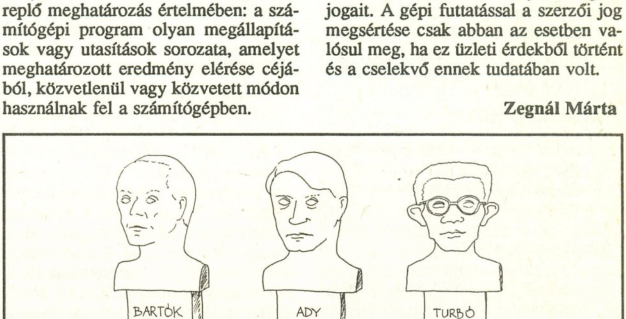
Japán

A japán szerzői jog védi mindazokat a számítógépi programokat, amelyek a törvény által előírt eredetiség követelményének megfelelnek. Nem részesül védelemben a programoknak az a köre, amely az iparban vagy egyébként közismert.

A program szerzője jogosult a neve feltüntetését kérni művének valamennyi példányán, erre irányuló jogát azonban a felhasználóval kötött szerződés korlátozhatja. A számítógépi program alkotójának a mű sérthetetlenségéhez való jogát a törvény behatárolja, amikor lehetőséget biztosít a jogos tulajdonos részére, hogy a program használatához feltétlenül szükséges változtatásokat és átdolgozásokat elvégezze.

A számítógépi programok saját célú, otthoni használata nem sérti a programok szerzőinek a törvényben foglalt jogait. A gépi futtatással a szerzői jog megsértése csak abban az esetben valósul meg, ha ez üzleti érdekből történt és a cselekvő ennek tudatában volt.

Zegnál Márta



Magyar szerzői jogszabályok

A szoftver hármas értelmezése

A hatályos magyar szerzői jog
a szoftvert három összefüggésben tárgyalja:
— mint szerzői művet.
— mint másodlagos szerzői alkotást.
— mint munkaviszonyban, munkaköri kötelezettségként alkotott művet.

A szerzői jogi törvény 1983-as végrehajtási rendelete a szoftvert mint művet lényegében a Szellemi Tulajdon Világszervezete 1976-os ajánlására visszanyúló módon határozza meg szerzői műként: a törvény hatálya alá esnek a számítógépi programok és a hozzájuk tartozó dokumentációk (a továbbiakban: szoftver). A szoftver szerzői alkotásnak minősítésével természetesen együtt jár a szerzői jogi törvény egész szabályrendszerének alkalmazása a programokra és a hozzájuk tartozó dokumentációkra. Ez az alkalmazás egészében véve akadálytalan, néhány ponton mégis értelmezésre és szerzői jogunk legközelebbi felülvizsgálatánál módosításra, kiegészítésre szorul.

Hézag vagy határozatlanság

Egyetlen példa a máris alkalmazható, de a szoftver esetében további értelmezésre szoruló szabályok közül: „Ha a szerző a mű felhasználásához hozzájárult, a felhasználáshoz elengedhetetlen vagy nyilvánvalóan szükséges, a mű lényegét nem érintő változásokat köteles végrehajtani; ha e kötelezettségének nem tesz eleget vagy nem tud eleget tenni, a felhasználó a változtatásokat hozzájárulás nélkül is végrehajthatja." (Szjt. 30.).

A második összefüggésben a jogszabály rögzíti, hogy a szoftver átírása, fordítása az eredeti nyelvtől eltérő programnyelvre, másodlagos szerzői alkotásnak minősül. Az 1969. évi III. szerzői jogi törvényből az is egyenesen következik, hogy a munkaviszonyban, munkaköri kötelezettségként alkotott szoftver felhasználási jogai átszállanak a munkáltatóra, amikor a szerző azt a munkáltatónak átadja. Az átadás a mű nyilvánosságra hozatalához való hozzájárulásnak minősül. A munkáltató jogosult a mű felhasználására a munkaviszony tartalma által meghatározott körben a működési területén. Ez a jog időben korlátlan, a teljes szerzői jogi védelmi időben fenáll. A munkáltató jogosult természetesen a szoftver saját célú, belső használatára (futtatására), valamint az ilyen szoftvertermék programkönyvtári elhelyezésére is. Ez utóbbi magyarázatot jogszabályi jogértelmezésként 1988-ban iktatták be.

Aki diktálhat

Az utoljára említett, harmadik összefüggésben, vagyis a munkaviszonyban munkaköri kötelezettségként létrejött szoftvereknél a végrehajtási rendelet külön meghatározza azokat az arányokat, amelyeknek megfelelően a munkavállaló — a szerző — a munkabér mellett részesedésre is jogosult abból jogdíjbevételből, amelyet a munkáltató a szoftverrel elért. Tekintettel az általában magas szoftver-előállítási költségekre, a munkavállaló szoftverszerző anyagi részesedése az egyéb művek szerzőihez képest alacsonyabb mértékű. Ez a mérték az általános 60-80%-kal szemben a szoftver esetében 10-30%. Megjegyzendő, hogy ugyanúgy, mint az egyéb szerzői mű kategóriáknál, a munkáltató itt is megállapíthat az alsó határnál kisebb arányú részesedést. Akkor teheti ezt, ha a feladatkörébe tartozik, hogy a műre harmadik személlyel felhasználási szerződést kössön, és a munkáltatónak a mű alkotásával kapcsolatban jelentősebb ráfordításai merültek fel.

A PTK és a bíróságok

A magyar szerzői jog a felhasználási szerződések általános szabályairól viszonylag részletes előírásokat tartalmaz, és egyúttal visszautal a Polgári Törvénykönyre is mint mögöttes jogterületre. A szoftverfelhasználási szerződésekről nincsenek olyan további, miniszteri szintű jogszabályban kifejtett előírások, mint például a könyvkiadásról, megfilmesítésről stb. Így a szoftverszerző és a felhasználó a felhasználási szerződések általános szabályainak keretében és figyelemmel a Polgári Törvénykönyv szerződésekről szóló szabályaira is, szabadon állapíthatja meg a szerződés tartalmát, beleértve természetesen az árat is.

A számítógépi programok szerzői jogi védelmének kapcsán a jogértelmezési kérdések tisztázásához a magyar bíróságok is jelentősen hozzájárultak. A legfontosabb döntések az alábbi kérdésekre irányultak:

1. A számítógépi programok tervezésének, elkészítésének egyes önálló szakaszai, illetve ezek eredménye szoftverként védett akkor is, ha utóbb ezek nem — vagy pedig másképpen — jelennek meg a tényleges programban.
2. A munkavállaló szerzői jogdíj-részesedésének alapja és kiszámítása.
3. A szerzői jogi szabályok elsőbbsége a polgári jog általános szabályaihoz képest a felhasználási szerződések körében.
4. A „munkáltató szokásos feladatköre" fogalom értelmezése.
5. A szoftver jogosítatlan átdolgozójának jogai abban az esetben, ha az átdolgozást harmadik személy felhasználja.

Gyertyánfy Péter

A VONATKOZÓ JOGSZABÁLYOK

— 1969. évi III. törvény és ennek végrehajtását szabályozó, 9/1969. (XII. 29.) MM számú rendelet.

Ennek módosításai:

— 15/1983. (VII. 12.) MM számú rendelet.
— 18/1988. (VIII. 24.) MM számú rendelet

A programozók (Artis)jussa

Az 1970-es évben a Szerzői Jogvédő Hivatal (Artisjus) felismerte, hogy a szoftverüzlet egyik alapvető feltétele — sőt, hosszú távon létéhez elengedhetetlen — a nemzetközileg is működő, kizárólagos jogi védelem. Ennek érdekében dinamikusan közreműködött a megfelelő intézményrendszer, valamint a jogszerű licenciagyakorlat kiépítésében.

A szoftverszerződésnek mint sajátos típusú szellemi alkotás forgalmát szabályozó megállapodásnak fő tárgya az a magatartás, amellyel a szerző a szoftver előállítását vállalja vagy annak használatát másnak engedi át. A szoftverszerződések alapvetően két csoportba sorolhatók: úgymint a jövőben alkotandó szoftverre vonatkozó — szoftverfejlesztési — megállapodások, továbbá a már meglévő szoftvermű felhasználásáról szóló — szoftverhasznosítási — szerződések.

Az első csoportba tartozó szerződések lényegében programozói kapacitás értékesítését teszik lehetővé meghatározott feladat megoldása érdekében. Az ilyen megállapodás tartalmazza a feladat pontos meghatározását, a kidolgozással szembeni követelményeket, rendelkezik a megalkotandó szoftverre vonatkozó felhasználási jogokról, a teljesítés módjáról, határidejéről, annak elfogadásáról. Megjelöli a felhasználó kötelezettségeit és az átadandó szoftver ellenértékét.

A szerződő felek megegyezhetnek a teljesítést elősegítő együttműködés kérdéseiben is. A szerződés meghatározza az esedékesség időpontját, a kifizetés feltételeit, az esetleges szerződésszegés mibenlétét, a szerződés felbontásának lehetőségeit, rendelkezik a jótállás időtartamáról. Több szerző esetén rögzíti a jogdíj teljesítéssel arányos felosztását.

A már elkészült szoftverek hasznosítását célzó ún. felhasználói szerződések rendelkeznek egyrészt az egyedi felhasználói szoftver hasznosításáról, másrészt tömegszoftver sokszorosított példányainak értékesítéséről tárgyi kódként (vagyis csak mágneshordozón, gépi jelekkel rögzített formában), továbbá a számítógép belső működését szolgáló rendszerszoftverek értékesítéséről géppel együtt .

A szoftverszerződések további szempontok szerint is csoportosíthatók. A nemzetközi szerződésekben például nagy jelentősége van a területi hatály kérdésének. Fontos a felhasználási jog időleges vagy végleges átadása, kizárólagossága, átruházható, illetve nem átruházható volta.

A nyolcvanas évek végéig Magyarországon nem beszélhettünk igazi szoftverpiacról. Ennek okát a keresleti oldal hazai jellemzőiben kereshetjük: a potenciális felhasználók — vállalatok, intézmények — szervezeti felépítési és működési szabályzatbeli sokszínűsége nem támasztott olyan igényeket, hogy a piacon többször értékesíthető szoftverek jelenjenek meg. Ebben az időszakban az Artisjus Ügynökség elsősorban szoftverfejlesztési szerződéseket kötött egyedi problémák megoldására. Ezt a tendenciát erősítette a hazai géppark heterogén jellege is; többször értékesíthető szoftverek elvétve bukkantak fel szerződések tárgyaként.

A számítástechnika itthoni fejlődésének íve a nyolcvanas években az ügynökség által kezelt szerződések mennyiségében is megmutatkozik. 1984-hez képest 1989-re az Artisjus-nál megkötött szoftverfejlesztési szerződések száma megduplázódott, és 1989-ben meghaladta a 17 ezret. Ugyanez olvasható ki a hazai és a külföldi felhasználókkal kötött szerződések arányának alakulásából is; amíg a nyolcvanas évek közepén a külföldi szerződések a belföldieknek mintegy három százalékát tették ki, az évtized végére ez az arány megközelítette a tíz százalékot.

Az utóbbi két év változásai gyűrték fel a magyarországi számítástechnika új vonulatait is. A gazdálkodó szervezetek átalakulása, a homogénebb géppark kialakulása mindinkább abban az irányban hatott, hogy az egyes felhasználók egyre ritkábban engedhették meg maguknak a luxust: korántsem mindig sajátos feladataikat máshol nem tapasztalt, egyedi problémaként értelmezve testreszabott szoftver fejlesztésével oldják meg. Be kellett látniuk, hogy addig speciálisnak tekintett helyzetük általánosítható, s ezáltal az ugyanabba a kategóriába sorolható feladatok elvégzéséhez elkészített szoftver nyilvánvalóan sok vevőnek adható el, s így a többször értékesíthető szoftverek gazdaságosabbak. Gyakorkatunkban ez a kész szoftverek felhasználását célzó szerződések szaporodását eredményezi, míg az „egyedi problémás" szoftverfejlesztési szerződések látványosan csökkenek. Ezzel egyidőben — noha céltudatos marketingtevékenység által is — szépen emelkedik a külföldi megrendelőkkel kötött megállapodások száma. 1990-ben, míg a belföldi szerződések növekménye 1989-hez képest nem volt számottevő, a külföldiek megkétszereződtek.

Az Artisjus ügyleteiben is érezteti hatását a számítástechnika fejlődési ütemének ez évi megtorpanása. A felhasználói oldal pénzhiányát mutatja a keresletcsökkenés, aminek következtében felértékelődött az ügynökségi marketingmunka: az évek során kialakított hazai és nemzetközi üzleti kapcsolatrenszernek köszönhetően az ügyfelek termékei számára piacot lehetett találni.

A gondok egyik kísérő tünete, hogy egyre több a szerződésszegés, ezért mind nagyobb súlyt kap az ügynökségi jogszolgálat. Ez év első felében annyi peres képviseleti ügy volt, mint egész 1989-ben.

Az állandó ügyfelek a partneri viszony egyik legnagyobb előnyének éppen a jogbiztonságot tartják.

A kedvezőtlen tendenciák mellett tovább fokozódik a jelentősége, hogy elősegítsük: a szoftverszerzők, -gyártók, -felhasználók és a nagyközönség jogos érdekei váljanak kiegyensúlyozottá és egymást erősítve jussanak érvényre. Vagyis: a szoftver jogvédelmét a felhasználások külföldi tulajdonosokat, befektetőket is megnyugtató jogszerűségével lehessen garantálni. Erre az évek során az Artisjus közreműködésével kötött, kereken 54 ezer szerződés tapasztalata a biztosíték.

Várkonyi Ágnes

A szoftverszerzőség vélelmezése és bizonyítása
A bírósági gyakorlat

A szerzői jog azt a személyt illeti meg, aki a művet megalkotta. Ez a személy a szerző, aki követelheti, hogy e minőségét senki se vonja kétségbe. Ebből a szabályból az következik, hogy szerzőként mindaddig a művön feltüntetett személyt kell tekinteni, amíg ennek ellenkezőjét nem bizonyítják. A bizonyítás terhe azon van, aki a feltüntetett szerző jogait vitatja.

A szerzőség valódiságának, illetve az ellenkezőjének bizonyításában rendszerint két ténycsoport játszik döntő szerepet: egyrészt az alkotás eredetisége, önállósága, másrészt az időbeliség. Az időbeliség rendkívül fontos tényező, hiszen ezzel dönthető el, hogy a szerző az alkotását eleve nem vehette át egy később készült műből.

Általában a szerzői jog területén az időbeliség kényes kérdés, hiszen nincs olyan intézmény, amely közvetlenül lenne hivatott a mű keletkezésének időpontját igazolni. A szoftverek esetében valamivel könnyebb a helyzet, mivel „nincsen szoftver dokumentáció nélkül", így a dokumentáció keletkezésének időpontja eligazításul szolgálhat. Persze bárki nyugodtan hivatkozhat arra, hogy a szoftverjogban nincs semmiféle regisztrálási eljárás — mint ahogy például a szabadalmi eljárásban van —, így nincs mód az elsőbbség igazolására, de a dokumentáció átadása általában nyugtázva van kísérő levéllel, átvételi nyilatkozattal vagy más módon.

A hiba mint „ujjlenyomat"

Természetesen nincs védelem az „íróasztali fiókokban" őrzött szoftverekkel való visszaélések ellen — ugyanúgy, ahogy önmagában az ötletet sem tudja védeni a jog. Képzeljük csak el, hogy mennyire lehetetlenül nehéz volna bizonyítani, hogy egy elkészült és ragyogó játékszoftver alapötlete nem a szerzőé. Különben is: az elkészült mű már az övé!

Az eredetiség és önállóság megítélése, ha lehet, még bonyolultabb kérdés. Ki és mivel képes alátámasztani, hogy azt a bizonyos alkotást ő és csakis ő hozhatta létre?! Nem zárható ki az az abszurd eset, hogy ugyanazt a művet — még ugyanolyan formában is — egy másik szoftveres szakember szintén megalkotta. Azért erre ma már vannak persze módszerek, de ezek nem jogi eszközök, hanem a szerzők által kidolgozott és sajátos, a szoftverekbe beépített védelmi módok. Bármely apró hiba, tévedés alkalmas lehet arra, hogy a szerzői jogosultságot igazolja, hiszen ugyanazt a hibát általában nem képes két ember egymástól függetlenül elkövetni. (Természetesen ilyen peres eljárás kevés van, csak egyet-egyet lehet a bírói gyakorlatban fellelni.)

Pereskedés a végtelenségig

A legtöbb per jelenleg rutinügyként kezelhető, és általában a felhasználói szerződések körében keletkezett szerződésszegésekből származik. A szerződésszegést többségében a felhasználók követik el a szerzői jogdíj meg nem fizetésével vagy a késedelmes teljesítéssel. Ilyenkor a jól megkötött szerződés komoly segítség az igény érvényesítésénél.

Véleményem szerint hasznos a felhasználót már a szerződésben szankcionálni a késedelem esetére — sok esetben az egyidejű teljesítés is kiköthető lenne. Ajánlható egy jó, bár ezen a téren még nem alkalmazott — de más szerződéses viszonyokban, például az adásvételi gyakorlatban igen hatékony — módszer. Éspedig: a felhasználó a szoftver átvételével egyidejűleg helyezze letétbe a szerzői jogdíjat a szerződést lebonyolító Szerzői Jogvédő Hivatalnál vagy ügyvédi irodánál, és a jogdíj majd csak a mű átadását és tesztelését követően — akár csakis a felhasználó kifejezett rendelkezésére — legyen kifizethető. Ez a megoldás védi a szerzőt attól, hogy nem kapja meg a jogdíjat, de védelmet jelent a felhasználónak is arra az esetre, ha a teljesítés nem megfelelő, és a szoftver kijavítására vagy esetleg átdolgozására van szükség. Attól is megmenti ez az eljárás a szerzőket, hogy a felhasználó alaptalanul hivatkozzon a hibás teljesítésre, hiszen megszűnik az ehhez fűződő alapvető érdeke, nevezetesen a pénz kifizetésének hátráltatása.

Az adós fizet?

Tapasztalatom szerint a konkrét szerződéses ügyekben a lehető leghatékonyabb eszközökkel kell a szerzőket felvértezni az ellenérték ki nem fizetése ellen, mivel a pereskedés gyakran eredménytelen. Természetesen van esélye, hogy a pert meg lehessen nyerni — előbb vagy utóbb —, de arra már jóval kevesbé lehet számítani, hogy a jogdíj be is hajtható. Nem egy konkrét ügyben előfordult, hogy a Polgári Perrendtartásra hivatkozással az alperesnek sikerült a pert hónapokkal, akár fél évvel is elhúzni, mondvacsinált védekezésekkel fizetési kötelezettségét késleltetni. Kevés szerző van, aki megengedheti magának azt a luxust, hogy hosszú hónapokig huzakodjon, akár évekig is — az egyébként megérdemelt jogdíjáért.

A gazdasági életben tapasztalható nehézségek ezen a területen is éreztetik hatásukat, és itt fokozott felelősség hárul a jogászra. Igenis hatalmas a jelentőségük a jól megszövegezett, egyedi és a konkrét ügyre szabott felhasználói szerződéseknek. Gondoljuk csak el, hogy mennyivel nyugodtabban aludhat az a szerző, akit a szerződése — akár a jogdíj letétbe helyezésével, akár más módon — megvéd a fölösleges pereskedéstől, mint az, aki hónapokig fut a bíróságon a pénze után. Aki csak teheti, kerülje az előre gyártott szerződések egyszerű kitöltését, és forduljon szakemberhez! Egy autót is olyan adásvételi okmánnyal veszünk meg, amelyet arra

az alkalomra készítettünk, vagy egy bútor eladásához is általában arra az esetre szóló „papírt” csinálunk, nem beszélve az ingatlanügyletekről. Miért becsüljük akkor le a szellemi alkotásokat. Nem érdemelnek ugyanannyi figyelmet? Mindenkinek csak azt tanácsolhatom, hogy ne sajnálja az időt ahhoz, hogy gondosan előkészített, részletes szerződést kössön.

A tisztesség kedvéért

A felhasználói szerződések közül nem egyben olyan megállapodással találkozhatunk, ahol a szerző az egyszeri jogdíjon túl ún. royaltyban részesül, ami azt jelenti, hogy a felhasználó által eladott példányszámok után meghatározott százalékot kap. Ez általában azt is jelenti, hogy a szerző kizárólagos felhasználásra engedte át a szoftvert, és így sem ő maga, sem más személy azt nem forgalmazhatja. Sajnos olyan ügyet is tárgyalt már a bíróság, ahol a továbbforgalmazó felhasználó nem a ténylegesen eladott példányszám után számolta a jogdíjat. Ez ellen megint csak a szerződés védhet, hiszen meg lehet állapodni abban, hogy a szerző betekintést nyerhessen a forgalmazás elszámolásába.

A Szerzői Jogvédő Hivatalnál szerzett gyakorlatomból megismert ügycsoportban előfordult olyan eset, hogy a szerző értesítést kapott a felhasználótól, miszerint az elkészített szoftver sajnos nem forgalomképes, a reklámozás és egyéb a forgalmazással járó költségek nagysága miatt eladhatatlan, ezért royalty fizetésére nincs mód. Ezzel egyidőben a felhasználó a szerzőnek minimális összeget átutalt az előző évi forgalom után, és jelezte, hogy a szerző több pénzre nem számíthat. A szerző tudomásul vette a közlést, hiszen nem volt más választása, mint hinni a felhasználónak. Nem sokkal később egy számítástechnikai újságban olvasta, hogy a felhasználó cég az első helyen áll a forgalmazási listán, vagyis abban az évben ő forgalmazta a legtöbb terméket. Még a legkelendőbbek listája is fel volt tüntetve, és a szerző megdöbbenéssel olvasta, hogy az általa készített szoftver bizonyult a statisztikák szerint a legnépszerűbbnek és a legforgalomképesebbnek. Arra is fény derült, hogy az újság az adatot a KSH-tól kapta, s oda közvetlenül a cégtől jutottak az információk.

Persze, ilyen frappáns eset azért ritka, hiszen itt a bizonyítás szinte gyerekjáték volt. Gyakoribb, amikor a szerző csak sejti, hogy többet forgalmazott a cég, mint azt állítja. Ilyenkor általában hosszas levelezés veszi kezdetét, és csak ennek eredménytelensége esetén fordul a szerző a bírósághoz. Sajnos mindenkit ki kell ábrándítanom, aki azt hiszi, hogy a peres eljárás nminden problémára gyógyír. Azon túl, hogy az utóbbi években lényegesen megnövekedett a bíróság munkája (és ezért az ügyek lassabban haladnak), az a probléma, hogy egy-egy megnyert ügyben a végrehajtás elég esetleges. Azoknak a cégeknek egy jó része, amelyek nem fizetik ki a szerzői jogdíjat és perre kényszerítik a szerzőket, nem is tudnak fizetni, és végrehajtás címén legfeljebb a banknál lehet sorban állni, majd várni, várni, várni...

„Alkalmazó” és alkalmazott

Még egy gyakori problémáról feltétlenül említést kell tennem, és ez a munkaviszonyban készített szoftverek témaköre. A rendelkezések szerint a munkaviszonyban készített szoftverek után csak akkor jár jogdíj, ha az alkotást a munkáltató harmadik személynek értékesítette. Az értékesítés tényét azonban akkor is meg kell állapítani, ha a munkáltató a szoftvert, illetőleg annak használatát vállalkozási, kutatási-fejlesztési vagy más szerződés keretében adja át. Ilyen esetekben a harmadik személy által fizetett összeg nyilvánvalóan magában foglalja a szoftver használatának ellenértékét is. Sajnos, nem ritka, hogy a munkáltató a szerzői jogdíj kifizetését — a saját maga által készített és a munkaviszonyban általánosan használt ún. szoftverszabályzat szerint — feltételhez köti. Ilyen feltétel lehet pl. az, hogy a végzett munka az átlagos intézeti nyereségen túl többleteredményt tartalmaz. Jogszabályellenes minden olyan munkáltatói kikötés, ami a szerzői jogdíj kifizetését bármilyen feltételhez köti. A szerzői jogi törvény értelmében ugyanis csak a kifizetés határidejét lehet bármihez is kötni, de magát a fizetési kötelezettséget nem.

Végezetül: eddigi tapasztalataim alapján azt tudom tanácsolni mind a szerzőknek, mind pedig a felhasználóknak, hogy a fölösleges és költséges jogviták elkerülése érdekében már a szerződés előkészítésének szakaszában forduljanak szakemberhez. Megéri.

Rusz Mária

Az Európai Közösségek Szoftver Irányelve

Jogfejlesztés ott, ahová igyekszünk

A számítógépi programok jogi védelmének jövőjét Európában meghatározza az EK jogfejlesztő tevékenysége. Ennek formája a tanács most kiadott irányelve — 1991. május 14. OJ. 14591 Nr. L.122/42.
Az irányelv a tagállamok kötelező feladatává teszi, hogy azok belső jogukat eszerint alakítsák át.
A határidő: 1993. január 1.

Szükséges a harmonizálás

Jóllehet Magyarországon az irányelv végrehajtását még az EK-val küszöbön álló társulási szerződésünk sem teszi kötelezővé, távolabbi terveink miatt nyilvánvaló, hogy követelményeit szerzői jogunk felülvizsgálatánál figyelembe kell vennünk.

Néhány tagállam szerzői jogában semmiféle említés nem volt a számítógépi programokról mint védett művekről, és ez önmagában is piactorzító elem a szoftverforgalomban. A hatályos szerzői jogi védelem szintje is eltérő országonként, továbbá a védelmi idő 25 és 70 év között váltakozik. Sőt, esetenként az eredetiségnek (az alkotás színvonalának) a védelemhez minimumként megkívánt mértéke is különböző. Noha az irányelv a kontinentális szerzői jogi és az angol-amerikai szerzői jogi felfogás közötti több kompromisszumot is tartalmaz, elősegíti az európai szoftverszerzők érdekeit és az európai szoftvergyártás fejlődését, versenyképességét.

Az érdemi szabályok

Az irányelv első cikke szerint a számítógépi programokat a tagállamok joga — irodalmi művekként — szerzői jogi védelemben részesíti. Eszerint a Tanács elutasította a szerzői jogon belüli önálló műkategória/alfaj gondolatát. Ez a megoldás bizonyos mértékig ellentmondó, mert — amint a továbbiakban látni fogjuk — az irányelv ugyanakkor (főleg a szabad felhasználás tekintetében) mégis szoftverspecifikus szabályokat állapít meg. Így számunkra is kérdéses, hogy vonatkoznának-e a szoftverre az egyes nemzeti jogokban az irodalmi művekre fennálló és az irányelvben nem említett speciális szabályok. (Például az idézés, az átvétel, a kiadás szabályai.)

Véleményünk szerint a szoftver olyan értelemben „irodalmi mű", hogy minden védett alkotás vagy irodalmi, vagy művészeti, vagy tudományos mű. Ezek a kategóriák azonban igen átfogóak, egymást részben át is fedik.

Az irányelv első cikke a program fogalmára nem ad meghatározást, viszont állást foglal a védelemhez minimálisan szükséges eredetiség kérdésében. Elegendő, ha a program nem másnak a szellemi alkotása, és ezen felül semmilyen további (minőségi, esztétikai) követelmény nem támasztható. Az előirányzott védelem — összhangban a Berni Uniós Egyezménnyel — a bármilyen formában kifejezett programot, így az olyan előkészítő anyagot, dokumentációt is megilleti, amelyből a program megkonstruálható. Nincs viszont szerzői jogi védelem az alapelvekre, így a külső és belső csatlakozási pontok, felületek (az ún. interfész) elveire sem. Tekintettel az interfészek megjelenésében, kifejtésében egyre terjedő standardizálására, várható, hogy ezek inkább csak kivételes esetekben állnak szerzői jogi védelem alatt.

A 2. és 3. cikk szerint a program szerzője egy természetes személy mint alkotó (alkotók) vagy — ahol azt a tagállam joga lehetővé teszi — a jogszabályban megjelölt jogi személy. A munkáltató jogszerzésének módjáról az irányelv nem szól, így az lehet eredeti (mint az amerikai jogban), de lehet ún. törvényi jogátszállás is valamely természetes személy szerzőről (mint a magyar jogban). Ez a szabály nem zárja ki a munkavállaló szerző — a magyar jogban ismert — részesítését a programhasznosítás bevételéből. Elvetették viszont azt a korábbi javaslatot, hogy a jogátszállási szabályt kiterjesszék a megrendelt programalkotásokra is — nem követve az angol, amerikai mintát.

Kizárólagos jogok és korlátaik

A szoftver-jogtulajdonos kizárólagos jogait és ezek korlátait a 4. és 5. cikkek tartalmazzák. A nemzetközi szerzői jogi egyezményekből természetesen következik, hogy a szerzői mű — így a program — minden többszörözéséhez, lefordításához és átdolgozásához elengedhetetlen a jogtulajdonos hozzájárulása. Az új szabályozást elsősorban az tette szükségessé, hogy az egyezményekből, nemzeti jogokból nem egyértelmű, vajon az időleges többszörözés a szerzők kizárólagos jogainak körébe esik-e, és meddig számít a rögzítés, többszörözés időlegesnek.

A kizárólagos jogok felsorolását a nevesített kivételek követik. Az irányelv ezeket a szabad felhasználási eseteket kizárólag a jogszerűen felhasználási jogot szerzett személyek javára állapítja meg. Az irányelv szerint a szerző hozzájárulásától függ a program vagy programrész minden tartós vagy időleges többszörözése. A betáplálás, képernyőn való megjelenítés, futtatás, tárolás, átvitel akkor engedélyköteles, ha akár csak pillanatnyi többszörözéssel is jár. (A mai technika mellett mindig ez a helyzet, noha a nemzeti jogok és a bírói gyakorlat egyes specifikus eseteket elvileg másként is megítélhet.) Az említett jogokat a jogszerű felhasználó — az engedélyezett felhasználáshoz szükséges mértékben és célra — továb-

bi nevesített engedély nélkül is gyakorolhatja, hacsak ezt éppen a felhasználási szerződés nem korlátozza.

Ugyancsak a szerző kizárólagos joga a program fordítása, átdolgozása, továbbfejlesztése, aktualizálása, de a jogszerű felhasználót a szerződés eredeti céljára — ideértve a hibajavítást is — ezek a jogok is szabadon megilletik. A szerző, eredeti jogtulajdonos ezt a szabad felhasználást is kizárhatja szerződéses kikötéssel. Ennek (a nyilvánvaló anyagi érdekek miatt) még sokkal több tere, indoka lehet, mint a jogszerű felhasználással együttjáró másolások, többszörözések szerződéses korlátozásának.

Terjesztés

A programszerző harmadik típusú nevesített joga a példányok terjesztése. Ennek van egy, a magyar jog szempontjából újszerű, az EK-ban viszont már jól ismert korlátja: a szerző engedélyével egyszer forgalomba hozott példányokra a szerzői jogok a továbbiakban nem érvényesíthetők, azok „kimerülnek", kivéve a példányonkénti bérbeadási jogokat.

Feltétlen — és szerződésben sem kizárható — korlátja a programszerző kizárólagos jogának, miszerint a jogszerű felhasználó a programról biztonsági másolatot készíthet, továbbá azzal próbafuttatásokat végezhet abból a célból, hogy a működés megfigyelésével megismerje az alapvető megoldásokat, elveket. Ez a szabály (5. cikk /3/ bek.) a iparjogvédelemben már ismert „kísérletezési szabadság szabályának" megjelenése a szerzői jogban. Ez biztosítja a tudomány továbbfejlesztésének akadálymentességét.

A magyar szerzői jogban szükséges lenne - más műfajok miatt is - a bérleti jog kifejezett nevesítése. Értelmezésre, az irányelvvel összhangban álló kiegészítésre szorulnak továbbá a szabad felhasználásról szóló jelenlegi szabályok: a biztonsági másolat készítése, a belső felhasználási célú hibajavítás és a jogosított felhasználó által végezhető továbbfejlesztés tekintetében.

Dekompiláció és „fair use"

Sajátos szabad felhasználási esetkör az irányelvben a „dekompiláció" (6. cikk). Ez a szabály az irányelv egyik központi rendelkezése. A számítógépi programokat illetően egymásnak feszül két érdek. Az egyik: a programalkotó és gyártó őrizze meg a kizárólagos hasznosítás versenyhelyzetét (engedély nélkül más ne másolhassa, használhassa a művet). A másik: a terjesztést fokozza az egyéb programokkal, a többféle hardverrel való együttes felhasználás (kompatibilitás, interoperabilitás) elősegítése is. Ez utóbbi viszont a program mások általi, bizonyos fokú szabad megismerését, használatát is feltételezi. A kérdés különösen azóta jelentős, mióta gyakorlattá vált a programok terjesztése gépi kódként. A dologban a következő a nehézség: a csatlakozó részek szabad megismerése, a program szétszedése (dekompiláció), illetve mérnöki visszafejtése a tárgyi kódból a forráskód irányába (reverse engeneering) igencsak megkönnyíti a plágiumot (elemek, rutinok álcázó összekeverése stb.). A nemzeti szerzői jogokban mind ez ideig nincs erre külön szabály, és így érthető, hogy az irányelv kompromisszumos megoldása a szabad használat értelmezéséről hosszú viták eredménye volt. Az irányelv célja egyrészt a fejlődő beruházások preferálása, másrészt a programok minél teljesebb „interoperabilitásának" támogatása.

A 6. cikk a „dekompiláció" címszó alatt a program egy bizonyos célú többszörözéséről és a programkód formájának megváltoztatásáról szól. Az eredeti programinformációnak egy másik programban való, kompatibilitási célú felhasználásához ez a két, szerzői jogilag releváns cselekmény elengedhetetlen. A többszörözést és fordítást — szabad felhasználásként — csak az eredeti program jogszerű felhasználója hajthatja végre, ha a kompatibilitás eléréséhez szükséges információk egyébként nem állnak rendelkezésre. A másolás, átírás csak az interoperabilitás szempontjából érintett programrészekre terjedhet ki. Az említett „bizonyos cél" pedig csakis egy — az eredetitől függetlenül megalkotott — másik programnak az eredeti programéval egyező eredményessége lehet. A „kutatás" vagy a versenyképes termék előállítása önmagában nem indokolhatja a szabad felhasználást. A dekompiláló alkotónak szükség esetén azt is tudni kell bizonyítani, hogy a jogos cél eléréséhez nem volt elegendő a program működésének — az 5. cikk /3/ szerinti — megfigyelése, elemzése a próbafuttatás során.

Az eddigiektől elkülöníthető és a 6. cikk /2/-ben külön szabályozott kérdés, hogy a dekompilációval nyert és átírt információ mire használható fel. Csakis az önállóan kifejlesztett programnak az eredetivel való interoperabilitására, nem pedig a másoknak való átadásra vagy az eredetivel versenyző, hasonló programok előállítására.

Mindezt a részletező — a „fair use" precíz amerikai szabályozására emlékeztető —, bonyolult előírást egy általános rendelkezéssel vezetik vissza a Berni Uniós Egyezmény köreibe. Az Egyezmény 9. cikk /2/ cikkelyére való utalás a dekompiláció korlátjaként állítja egyrészt azt a veszélyt, hogy a jogtulajdonos kétségbevonhatatlan érdekei indokolatlanul sérülhetnek, másrészt pedig a program normális, szokásos hasznosításába való ütközést. Tehát a dekompiláció mint szabad felhasználás csak végső esetben megengedhető.Mind ez ideig a magyar szerzői jog sem ismer szabályt a reverse engeneeringre, dekompilációra. A szabály honosítása — legegyszerűbben a szó szerinti átvétele — már rövid távon is indokolt, távlatban pedig szükségszerű lesz.

A 7.-től a 11.-ig

A szankciókról szóló 7. cikk olyan jogsértéseket azonosít, mint amelyeket az amerikai jog „másodlagos szerzői jogsértésekként" szabályoz. Mindez a magyar szerzői jog szempontjából is újdonság. A szerzői jogi (polgári jogi) szankciók nálunk ma csak a közvetlen jogsértőre irányulnak. A jogsértő példányok ugyan most is elkobozhatók harmadik személyektől — akkor is, ha ők a példányokat jogszerűen szerezték meg —, de ez csak polgári jogi eljárás keretében (és speciális eseteknél a szabálysértési eljárás során) lehetséges. A jogtalan felhasználással véghezvitt jogsértés büntetőjogi szankciói még hiányoznak.

A védelmi idő kötelező minimumaként a 8. cikk a Berni Uniós Egyezmény szerinti 50 évet írja elő — a szerzők halálától, illetve ahol a jogi személyek eredeti jogszerzése elismert, a program nyilvánosságra hozatalától számítva. Azok a tagállamok, ahol a védelmi idő hosszabb, ezt a közösségi jog e tárgyú harmonizálásáig megtarthatják; azoknak, akiknél a védelmi idő ma rövidebb, 1992. végéig fel kell emelniük 50 évre.

A záró rendelkezések (9., 10. és 11. cikk) közül fontos a más jogintézmények — például szabadalom, szerződés — által adható, esetleges párhuzamos védelmi lehetőségek megerősítése. A szerződéses védelmi kikötés persze nem semmisítheti meg az interfészre vonatkozóan az irányelvben kötelezően fenntartott szabad felhasználási lehetőségeket. Az irányelv érvényes a hatályba lépés előtt készült programokra is (pozitív visszaható hatály).

Gyertyánfy Péter

Lopunk, szerzünk, adunk-veszünk...
Segíts magadon, a COPYRIGHT is megsegít!

Amióta vannak azonos felépítésű számítógépek, azóta vannak programlopások. Feltehetően mindig is lesznek. Nem szép dolog ez, de hát más tények is akadnak körülöttünk, amelyek csúfak és gonoszak... A szoftver-adásvételnek mindinkább fair play ügyletnek kell lennie — mindkét fél részéről.

A lopásnak csak egyik oka, hogy így kockázat nélkül — még a vétel előtt — ki lehet próbálni a terméket: elkerülve azt, hogy zsákbamacskát vásároljunk. A csalódásnak egyébként nem elhanyagolható a valószínűsége, mert a tényleges működési környezetbe telepített program sokszor mutat olyan tulajdonságokat, melyek a vásárlót meglepetésként érik. Ez persze azért következhet be, mert a reklám és az eladó mást ígért. Van tehát, aki nem mond igazat, és van, aki ezt időben kideríti. Egyik nincs a másik nélkül, kölcsönös visszaélés ez valamely helyzettel, illetve lehetőséggel. És végül is senkinek sem igazán jó.

Viszonyaink és viszonyításaink

A politikai helyzetnek, továbbá a CO-COM előírásainak komoly szerepe volt az eszközök elterjedésének megakadályozásában. Néha még komoly felárért is csak nehezen lehetett elcsípni valami jót. A vasfüggöny időszakában szinte dicsőségnek számított olyan programok becsempészése, amelyeket tiltott gyümölcsként jegyeztek, és ezeket az anyagokat hétpecsétes titokként őrízték — vásárolni is csak darabokat lehetett belőlük. (A szoftvercsempészek és -árusok komoly vagyonokat kerestek e tevékenységükkel.) Követésről, új verziókról szó sem lehetett. Sőt, ha a gyártótól az azonosítási szám birtokában érdeklődtünk, még akkor sem válaszolt. Ha egy termékről információkat kértünk, két évvel ezelőtt még sokszor kaptunk olyan válaszokat Amerikából, hogy nem kívánnak magyar cégekkel együttműködni.

A programok terjesztése három magatartási modell szerint alakult:

a) A megszerzett programokat másolták, esetleg módosították, majd pénzért terjesztették.

b) A programokat a felhasználók, szakemberek kopírozták egymás számára, de nem kértek pénzt e szivességért.

c) Ha valaki valamit megszerzett, nem adta oda senkinek.

A szakemberek között a b) modell terjedt el leginkább. A kereskedők közül sokan követték az a) modellt; így rengeteg programból készültek illegális példányok. Érdekes módon ez az illegális terjesztés nemcsak hátrányokkal járt a gyártók számára. Néhány szoftver óriási népszerűségre tett szert, és ezek legalizálása, illetve az új verziók eladása révén új piacok nyíltak a gyártók számára.

A mai magyar szituáció kialakulásában — például a PC-kategóriájú gépeken a Novell hálózatok ilyen mértékű elterjedésében — meghatározó szerepe volt a Novell könnyen másolható kulcskártyájának. Ha valahol Magyarországon összekötött PC-k működnek, nyugodtan fogadhatunk arra, hogy azokon a Novell valamilyen változata fut. Ma már mindenki jogos példányt akar használni, de nem lenne ennyire egyeduralkodó a Novell, ha komolyabb védelmi rendszere lett volna.

Övé — tied — enyém

Volt ennek az állapotnak néhány nagyon világos következménye. Az említett meggazdagodáson túl a programok tulajdonjogának tiszteletben tartására is komoly hatással volt az a zűrzavar, ami az ügyeskedésekkel járt. Még mindig érdekes kérdés azonban, hogy lehet-e védekezni a lopások ellen, és milyen módszerek ajánlhatók.

A szoftver történetét tanulmányozva az látható, hogy eleinte nem védték a programokat, mert nem folyt nagyarányú terjesztés. Ahogyan később terjeszkedett a számítástechnika, a gyártók igyekeztek megakadályozni a másolást. Ez elsősorban a PC-s éra korai szakaszára jellemző. Ma már fontossá vált ezen a területen is a bizalom, ezért a gyártók egyre kevésbé védik programjaikat különleges rafinériák által. A szoftverlopás megakadályozására vagy korlátozására kétféle megoldás járatos — mindkettő buktatókkal bár.

1. Valamilyen másolás elleni védelmet alkalmaz a gyártó; ezzel megakadályozza az illegális reprodukciót, birtoklást. Lehet ez hardver (például kulcskártya) vagy szoftver (például információelrejtés az adathordozón, melynek az a következménye, hogy a program nem működik több példányban).

2. Minden programpéldánynak egyedi azonosítója van; ennek alapján a gyártók regisztrálják vásárlóikat, és olyan szolgáltatásokat próbálnak nyújtani a bejegyzett felhasználóknak, amelyek értéke összemérhető a program árával. Számos, e taktikát követő cég előírja, hogy programja csak egy számítógépre installálható.

Az első esetben az lehet a legnagyobb gond, ha a védelmi rendszer felmondja a szolgálatot, vagy „összeakad" bizonyos programokkal és ezért a jogos tulajdonos a programot nem tudja használni. A második módszer hátránya teljesen kézenfekvő: az ilyen program tetszőleges számú példányban előállítható. Ezek a szoftverek persze tartalmaznak egyedi azonosítókat, regisztrációs kódokat; így a másolatok forrása kideríthető. Ez a módszer önmagában nem elég hatékony, ezért egyes gyártók alkalmazzák a kettőt együtt is. Figyelembe véve azonban az első módszerben rejlő problémákat, talán érthető, hogy mégis a második térhódítása figyelhető meg a gyártók között, és a felhasználók is előnyben részesítik az ilyen programokat.

A vásárló szemszögéből még azt is érdemes mérlegelni, hogy ha egy szoftver készítői és forgalmazói nem alkal-

maztak védelmet, de a követéssel és egyéb előnyökkel támogatják a regisztrált példányokat, mit is érnek majd ezek a folyományok az alkalmazás során. Kétséges lehet ugyanis, hogy a gyártó (és a kereskedő) valóban tud-e adni olyan plusszokat a felhasználó számára, amelyek értéke összemérhető a regisztrálható példány árával.

Megfigyelhető, hogy nálunk is a nyugaton bevált terjesztési formák terjednek: dealerek neveződnek ki, ezek igyekszenek megfelelően felkészülni, hogy érdemben legyenek képesek közreműködni a várható problémák megoldásában. A világban bekövetkezett gazdasági-politikai oldódás következtében a gyártók is nyitnak a felhasználók és egymás felé is. Nyílt rendszereket készítenek, hogy lehetővé tegyék a többiek csatlakozását az ő rendszereikhez. Ennek az új iránynak az alapja a UNIX operációs rendszer, amely a jelek szerint néhány éven belül uralkodó szerepre tesz szert a világ számítógéphálózatain. A UNIX hazai elterjedését jóformán lehetetlenné tették a COCOM előírásai, ezért az alkalmazott számítástechnikában — a nyílt rendszerek későbbi használatában — nehezen ledolgozható hátrányokkal kell számolnunk.

Hogy is, mint is?

Egy számítástechnikai cég gyakorlatából tekintve most vissza, személyes emlékként említhetem, hogy ezekben a homályosabb időkben mi is igyekeztünk komolyan védeni a magunk által kreált programokat. Jelentős erőfeszítésekkel kifejlesztettünk egy olyan védelmi rendszert, amely több géptípuson is akcióképes volt. A felhasználók általában ma már meg is követelik az eredeti anyagokat, de önmagukkal sokszor nem ilyen szigorúak: több példányban is használják a programokat. Mi is mindig megpróbálunk a gyártóval kapcsolatba kerülni, de ha nem sikerül, közvetítőktől is vásárolunk. Mindenképpen eredeti szoftvert veszünk. Előfordulhat, hogy a program nem a hivatalos disztribútortól származik. Ebből esetleg később lehet probléma (például a követéssel vagy az upgraddel), de a disztribútor sokszor nehézkes és drága.

A hazai gyártású programokkal szemben is változóban van a felhasználók magatratása. Komolyan igényt tartanak a követésre. Mi sem védjük már saját programjainkat, és nem tudunk arról, hogy illegális példányokat használnának belőlük.

Vincze Sándor

„Lopy Right"
A programlopás anatómiája

Az alábbi gondolatok a felhasználó szemszögéből, talán kicsit a „vágyálmok" felé elhúzva világítják meg a közelmúlt tapasztalatait és a kívánatos jövőt.

A számítógépes programok zöme — a szabadszoftverek (shareware, freeware, public domain) programok kivételével — elméletben hazánkban is valamiféle jogi védelmet élvez, amint arról a kísérőlemezes és nyomtatott dokumentumok egyértelműen tájékoztatnak. Nem ritkán bejelentkezésekor maga a program is ugyanezt teszi — hiába. A programok és leírások másolása — megint csak elméletben — szigorú feltételekhez kötött, mégis elenyésző a ténylegesen megvásárolt programok hányada a „megszerzett" programokéhoz képest.

Például: ugyan hányan vettük meg a géppel együtt a DOS-t? Vajon milyen lehet a megvásárolt és a — ne köntörfalazzunk: — lopott Norton Commander, PCTools, dBASE, Lotus, Word programok aránya, hogy a többi elterjedt szoftvert ne is említsem. Ezek között talán minden huszadik-ötvenedik szoftver nem esik a „lop-ware" kategóriába.

Miért és hogyan alakult, alakulhatott ki ez az áldatlan helyzet?

— Az általánossá vált garasoskodás miatt „dicséretes bűn" lett a vásárlás nélküli programbeszerzés. Legtöbbször az jár jól, aki pénzkiadás nélkül tudja bővíteni programkészletét.

— A Commodore-korszakból nemcsak örököltük, de tovább is fejlesztettük az „én is adok, te is adsz" játékot.

— A lop-ware helyzet miatt egyre sokasodó programvédelmek ezek dühödt feltörésére és a feltört verziók gátlástalan terjesztésére késztetik a méltán felbőszült hozzáértőket.

— Amíg egy mezei diplomás itthoni havi átlagbérének 2-, 4-, sőt 8-szorosába kerül egy-egy jobb program, addig magánemberként kevesen fogják azt megvenni; a vállalatnak pedig olcsóbb egy alulfizetett diplomással elvégeztetni azt, amit egy drága program pár perc alatt elvégezhetne — akár egy titkárnő keze alatt.

— A szoftverforgalmazók jelenlegi üzletpolitikája is hibáztatható. Amíg a regisztrált program tulajdonosa szinte semmi pluszt nem kap a pénzéért, addig ugyan mi késztesse a vásárlásra?

Mivel sem a programok árának drasztikus csökkenése, sem a vásárlóerő növekedése, sem pedig a morál javulása magától nem várható, a programvédelmeket pedig majd a piac spontán alakulása fogja csak megszüntetni, az egyetlen járható út a megvásárolt szoftverekkel járó kiegészítő szolgáltatások jelentős bővítése.

Tudom, nem két fillér egy szoftvertanácsadó szolgálat fenntartása, de nem ártana összehozni egy olyan irodát, ahol összegyűlhetnének a különböző programok szakavatott ismerői. Akár olyan módon, hogy a szoftverforgalmazó cégek közösen szponzorálnák a tanácsadó irodát, ellátnák információval, hogy ki mit vásárolt meg és mire jogosult. Először természetesen csak néhány tucat program ismertetését, támogatását végezhetné az iroda, a támogatott programok száma végül a kereslet/kínálat függvényében alakulna. Ilyen formán hosszabb távon mindenképp megtérülne ez a befektetés. Az sem mellékes, hogy ez az iroda független beszerzési tanácsadóként, sőt referenciahelyként, valamint bemutató és oktató bázisként is szolgálhatna, ami mindenképp költségmegtakarítást eredményezhetne a szponzoroknak, nem beszélve a felhasználók hasznáról.

Az irodát természetesen úgy is el tudom képzelni, hogy a ténylegesen ott dolgozók az érdeklődőknek megadják az adott program(ok) szakértőjének a telefonszámát. Ezek a szakértők akár másodállásban is végezhetnék a tanácsadói tevékenységet fő munkahelyükön vagy otthon, s így még a terem- és gépköltség is tetemesen lecsökkenne. Ez a megoldás lehetővé tenné, hogy 2-12 órán belül bármilyen gond kiküszöbölésére találnánk megfelelő szakértőt. A bedolgozó kollégák honorálását átalánydíjas vagy eseti szerződésekkel kellene rögzíteni. Mindenesetre várjuk a reflexiókat és az érdeklődőket.

Nagy Gábor

Érdekeink, érdekeltségeink

Ami jogtiszta, az drága?

Magyarországon a számítógépes kultúra az elmúlt évek gazdasági és társadalmi körülményein felülemelkedett, és a hasonló helyzetű (volt KGST) országok színvonalát meghaladja. Számos szoftveres szakemberünk dolgozik külföldön, és sok hazai cég pályázik nyugat-európai szoftverfeladatok elnyerésére.

Az európai integráció jegyében új kihívásokra kell reagálnunk, kilépve a jelenlegi gyakorlatból, ütköztetve azt a nyugat-európai, illetőleg amerikai követelményekkel.

Az alábbiakban egy szoftveres kötetlen elmélkedéseit olvashatják, melyet 25 éves, főként hazai szakmai gyakorlat után vetett papírra.

A jó szoftvernek egyrészt fizetőképes kereslete van a tőkés piacon, másrészt azonban nagy a verseny és temérdek kiváló szoftvertermék jelenik meg. Sürgős feladat tehát részint megismerni azokat az új formulákat, amelyek használata rendszeresen jelentkezik a versenytárgyalások alkalmával, részint olyan minőségű, egyedi szoftvertermékeket készíteni, amelyek gyártási színvonala kielégíti a nyugati igényeket.

A szoftvergyártás

A szoftver szellemi termék és áru, amit ma már elismer a társadalom. Ez az áru szorosan összefügg a műszaki haladással, ezért elavulása, használati értékének idólegessége, valamint rövid életciklusa sem vitatott. Lényeges sajátossága a gyártási színvonal, a megbízhatóság és az árukra jellemző, számos gazdasági, kereskedelmi mutató, mint a piacképesség és a jogvédelem.

A hazai szoftvertermékek szervezett minősítése, illetve ennek alapján történő osztályozása esetleges. Néhány szoftverház belső értékeléssel kívánja termékeinek színvonalát biztosítani, de ez a módszer és az értékelés szempontrendszere még nem elég kidolgozott, nem nyilvános.

A kitűzött célt, a versenyképes gyártást a piac szabályozó hatása nagy késéssel követi.

A szoftveres képzésről

Napjaink szinte összes nagy szoftveres cége egymással versenyezve saját egyetemi programmal rendelkezik, melynek keretében rendkívüli mértékű, de rendszeres árkedvezményekkel (50-80%), az egyetemek elárasztásával kívánja megerősíteni termékeinek hosszú távú piacát. Ez a gyakorlat hazai berkekben is terjed, és a kellő mértékű kölcsönös készség igen hamar kialakul. A termékek tartalmának, használatának és környezetének a megismertetésére, a szoftver- és a gyártási kultúra emelésére — véleményem szerint — ez a módszer igen jó lehetőséget ad.

A szoftverforgalmazás

A szoftver mint áru szorosan kapcsolódik a csúcstechnikához, ezért stratégiai, politikai jelentősége lényeges. A hazai politikai változások ütemében és mértékében hozzájuthatunk olyan nyugati kereskedelmi termékekhez, melyek az aktuális COCOM-előírásoknak megfelelően legálisan beszerezhetők. A csúcstechnikához közeli szoftverek jogvédelme a nyugati országokban még lényegesen eltér a hazai gyakorlattól. Jogrendünk minden bizonnyal követni fogja az e téren ránk váró korszerűsítést.

A szoftveres társadalom azonban jelenleg Magyarországon nehezen alkalmazkodik az új szokásokhoz, követelményekhez, és ezzel lassítja saját termékeinek reális piaci értékelését. Ugyanis egy önálló hazai szoftver, amely drága (jogtiszta) nyugati szoftverek alkalmazásával készül, automatikusan magasabb kategóriába lép. Ilyen irányba hat az alkalmazott alapeszköz ára és annak hatékonysága.

A jogvédett nyugati szoftverek hazai forgalmazóinak egy része megkísérli áthidalni a még jelentős szakadékot, és a vásárlókat szerződéssel készteti a megfelelő gyakorlat elfogadására.

Mivel a szoftvervédelem egyéb módozatai (vírus, hardverkulcs stb.) rendszeresen sok nehézséget okoznak, a komoly cégek, melyek a felhasználóval való kölcsönös megbecsülés és bizalom alapján folytatják kereskedelmüket, ilyen módszerekhez nem folyamodnak.

A jogtiszta szoftver

Hogy a korszerű piac miként dolgozik, illetve hogy a modern kereskedelem és az érdekeltség összhangja mit takar, egy példával világítható meg.

Vegyük szemügyre az Informix, egy amerikai gyártmányú, a világ élvonalán mozgó, relációs adatbázis-kezelő termékcsoport hazai forgalmazásának „kellékeit".

— Minden termék gyártási számmal van ellátva, amely alapján a gyártó cég és a gyártási idő azonosítható, minden példány sorsa követhető.

— A cég a termékei fejlesztésének eredményeként elkészült újabb verziókkal a nyilvántartott jogos felhasználókat rendszeresen és költségmentesen ellátja.

— A jogtiszta termékek tulajdonosai minden kereskedelmi akcióban, illetve az igényelt szakmai támogatás során lényeges előnyöket és kedvezményeket élveznek.

— A jogos tulajdonosokat a cég és képviselői disztribútorokon és felhasználói körökön mint szakmai érdekképviseleteken keresztül látja el a legfrissebb információkkal és segédeszközökkel.

A végfelhasználók és a fejlesztők viszont szerződés alapján kötelezik magukat a termékre vonatkozó előírások betartására, mely ebben a viszonylatban egyértelműen érdekük is. (Nem ilyen tartalmú, de mégis aktuális szerződésmintát mutatunk be olvasóinknak a 17. oldalon.)

Csillagh Péter

SZOFTVERKÖLCSÖNZÉSI SZERZŐDÉS

Létrejött a .... (cég és cím), valamint .... (a továbbiakban ügyfél) között. A szerződő felek megállapodnak az alábbi szoftvertermék kölcsön formájában történő átadásáról-átvételéről és használati jogának a továbbiakban részletezett módon történő rendezéséről:

A termék neve: ....
Beszerzési ára: ....
Azonosítási száma: ....
SW kulcs kódja: ....
Verzió száma: ....
Operációs rendszer: ....
A hálózat típusa: ....
Felhasználók száma: ....

Az ügyfélnek tudomása van arról, hogy:

— A kölcsönzött szoftver a .... (cég) tulajdonát képező, eredeti gyártmány.

— A szoftvertermék, amelyet a (cég) kizárólag az ügyfél számára enged át, hiánytalan és jogtiszta.

— Az átadás a termék tanulmányozása, teljesítményének vizsgálata céljából történik, és abba betekintést kizárólag az ügyfélnek engedélyezik.

— A (cég) a fenti szoftverváltozatot az ügyfélnek csak az alábbi licencfeltételek alapján, kölcsönképpen és meghatározott időre engedi át.

I. SZOFTVER SZERZŐI JOGOK, LICENCEK

1. E szoftver kölcsönvételével az ügyfél nem szerzi meg a fenti változat tulajdonjogát, hanem csak meghatározott időre szóló használati jogot szerez.

2. Az ügyfél kizárólag egy adott számítógéprendszeren (az engedélyezett számítógéprendszer) jogosult a neki átengedett változat használatára. Ebből a szempontból a hálózatban összekapcsolt számítógéprendszerek nem tekinthetők egy rendszernek. A programok más számítógéprendszeren történő használata tilos. Amennyiben a változatot hálózatban való használatra engedik át, akkor a használati jog csak a (cég) által nyilvántartott és a szoftver gyártója által engedélyezett konkrét munkahelyekre vonatkozik. Az átengedett szoftvert az engedélyezett számítógéprendszerről nem szabad elektronikus úton más számítógéprendszerre vagy hálózatra átvinni.

3. Az átengedett változatokat az ügyfél sem gépi kódban, sem nyomtatott formában nem másolhatja le, még biztonsági okokból sem. A felhasználói kézikönyvről sem szabad másolatot készítenie. A szoftver harmadik félnek nem adható át. Ebből a szempontból harmadik félnek számítanak az ügyfél fióküzletei és leányvállalatai is.

A szoftver gyártójának vagy a (cégnek) a programban vagy a felhasználói kézikönyvben található jelzéseit, Copyright jeleit és tulajdonosi megjelöléseit az ügyfél semmilyen módon nem változtathatja meg.

II. TÉRÍTÉS

A változat átengedése:

a) Ingyenes.

b) .... Ft térítés ellenében történik.

III. HASZNÁLATI IDŐ

A szoftver a (cég) általi átadástól, illetve a szerződés aláírásától számított ... hónapig áll az ügyfél rendelkezésére. A használati idő leteltével köteles azt újból a (cég) rendelkezésére bocsátani, illetve kérésére képviselőinek jelenlétében megsemmisíteni.

IV. JÓTÁLLÁS

Az ügyfélnek tudomása van arról, hogy az átvett változat teljes értékű szoftvertermék, amelyet kizárólag tanulmányozás céljára lehet felhasználni. Egyéb felhasználása az ügyfél saját kockázatára történik. Az ügyfél csak abban az esetben léphet fel követeléssel, ha a (cég) részéről szándékos károkozás történt, minden más esetben az ilyen követelés kizárt.

V. FELELŐSSÉG

Az ügyfél tudomásul veszi azt a nyomatékos figyelmeztetést, miszerint a kapott szoftverrel adatfeldolgozást nem végezhet. Amennyiben próbaképpen adatokat akar vele feldolgozni, azt csak akkor teheti, ha azok elvesztése esetén kár nem keletkezik.

Az ügyfél a fenti termék megsemmisüléséért vagy a szerződés megszegéséért a fenti beszerzési ár mértékéig felel.

VI. EGYÉB KIKÖTÉSEK

1. A mindkét fél részéről történő szolgáltatások helye:

....

Az ügyfél és (cég) között felmerülő vitás kérdésekben elsőfokú bíróságként a .... Bíróságot (cím) jelölték ki.

2. Arra az esetre, ha az ügyfélnek az adott helységben nincs általános illetékessége, vagy ha a szerződés megkötése után székhelyét vagy szokásos tartózkodási helyét Magyarország területéről máshová teszi át, akkor a szolgáltatás költsége a helyszíntől függően megemelkedik. Amennyiben az ügyfél szokásos tartózkodási helye nem ismeretes, a szerződő felek (közigazgatási egység) területi hatáskörében állapodnak meg.

3. A (cég) és az ügyfél közti jogi kapcsolatot kizárólag az MK jogi előírásai szabályozzák.

4. Kiegészítőleg érvényesek még a (cég) általános üzleti feltételei és a mindenkori megbízások igazolásai.

5. Amennyiben a jelen szerződés egyes rendelkezései nem lennének hatékonyak vagy valamely később felmerülő körülmény folytán elvesztenék jogi érvényüket, vagy ha ebben a szerződésben bármilyen hiányosság merülne fel, mindez nem érinti a többi rendelkezés jogi érvényét. A szerződés érvénytelen rendelkezései helyett, illetve a hiányosságok pótlására megfelelő szabályozásnak kell érvénybe lépnie, amelynek (amennyire lehetséges) minél jobban meg kell közelítenie a szerződő felek feltételezett eredeti szándékát.

(Keltezés)

aláírás (cég) aláírás (ügyfél)

Egy logikus lépés...
Gyártófüggetlenség és nyílt rendszerek

A „Copyright" kapcsán érdemes körülnézni annak tágabb fogalomkörében is, kitekintve például a szabványosítás tendenciáira. Egy látszólag logikátlan állítást szándékozunk alátámasztani: ami a legújabb, legkorszerűbb és leghatékonyabb, az korántsem mindig eredeti...

Egyre többen ismerik fel itthon is, hogy nagyobb számítógépi teljesítményre van szükségük. A nyiladozó piaci rést azonnal érzékelve, ilyen gépek kínálatával elsőként a nagy külföldi cégek jelentek meg: IBM, DEC stb. Kiváló minőség, megfelelő teljesítmény, borsos ár — így jellemezhető tömören a kínálatuk. Illetve, ami az árat illeti, első látásra az nem is olyan sok. Egyre gyakrabban kínálnak használt, néhány éves, uraságtól levetett konfigurációkat bérleti szerződések keretében. Ezek szinte mindig az adott cég — noha bizonyos kategóriákban szinte világszabványt képviselő — speciális operációs rendszerével működnek, és kerülik a magasabb szinten szabványos, nyílt rendszerekre jellemző megoldásokat.

Kiszolgáltatottság

Hatalmas csapda ilyen gépet vásárolni! Éppen azért lehet ilyen rendszerekből egy-két éves „second hand" gépet venni, mert ezek a fejlett ipari államokban tulajdonképpen megbuktak: nincs hozzájuk elegendő olcsó szoftver, és a speciális, gyártóspecifikus megoldások miatt a felhasználó ki van szolgáltatva a gyártó kénye-kedvére. A gyártók ezzel a „futok a pénzem után" jellegű felhasználói kényszerhelyzettel számolva, pozíciójukat szemérmetlenül kihasználják: a hardver- és szoftverbérleti díjak, karbantartási átalánydíjak, vételi törlesztőrészletek rendkívül magasak, és előre rendszerint nem is ismertek.

A szocializmus máról holnapra élő pazarlásában felnőtt nemzedék számára kicsit idegen, hogy néhány évre előre vegye figyelembe a költségeket, de a piacgazdaságban ez elkerülhetetlen. Tudomásul kell venni, hogy az informatikára fordított beruházás a vállalkozás beruházásainak, esetenként legnagyobb részét teszi ki, és szerencsétlen megválasztása — még ha műszakilag meg is felel a követelményeknek — tönkreteheti magát a vállalkozást is.

Szabványos felületek

Kiutat a PC-hálózatok és a gyártóspecifikus rendszerek csapdáiból az ún. nyílt rendszerek kínálnak. Más, idősebb iparágakban a szabványosítás jelentősége óriási; míg a számítástechnikában az elképesztően gyors fejlődés eddig háttérbe szorította azt. Mire egy szabvány megszületett, már el is avult. Most, hogy ez az iparág is korosabb — noha fejlődésének lendülete még mindig töretlen — sikerült megtalálni azokat a szabványokat, amelyek nem fékezik a fejlődést, hanem az egységesség miatt elérhető magasabb gyártási darabszámok jóvoltából éppenséggel serkentik. Ezek a szabványok elsősorban interfészek: szabványos felületek rendszerek és rendszerelemek között. Nem kell tehát teljes számítástechnikai rendszert előállítani gigantikus cégeknél, elegendő csak egy-egy szabványos interfészű elemet gyártani — vagy ilyen szabványos elemekből egy rendszert összeállítani.

Manapság a szabványok mennyisége és a szabványok alapján gyártó vállalkozások száma elért egy kritikus tömeget. Ennek eredménye egy nyugaton viharosan terjedő irányzat: az informatikát ilyen nyílt, szabványos rendszerekre alapozzák.

A nagy teljesítményű eszközök körében naponta dőlnek halomra a csúcsok: egyre kedvezőbb ár/teljesítmény mutatóval jelennek meg a rendszerek. Ugyanolyan földcsuszamlásnak vagyunk a tanúi, mint amilyet a PC-k jelentettek korábban.

A nyílt rendszerek szabványai egyre szélesebb körben hálózzák be az iparágat. Jelenleg az operációs rendszer szabványosítása áll a középpontban. Ennek jelentősége óriási: nagymértékben függetlenítheti az alkalmazói szoftvereket a hardvertől. A jelenlegi elképzelések szerint az alkalmazói programok már a nagyon közeli jövőben bináris kód szintjén lesznek átvihetők az azonos utasításkészlettel rendelkező processzorok között.

Egyetlen jelölt: az UNIX

A nyílt operációs rendszer alapja az egyetlen lehetséges jelölt, a UNIX lett. Kisebb viták még folynak, de a legfontosabb gyártók gyakorlatilag megegyeztek. Jelenleg még nem teljes a szabvány, de 1-2 éven belül komplett lesz; és akik UNIX-platformról indulnak a változáskor, azok minimális módosítással érhetik el majd a kínált lehetőségeket.

A hardver területén néhány régebbi és újabb szabvány szinte teljessé tette a gyártótól való függetlenséget. Már a kaliforniai számítógépgyártás legtöbb cége is a piacon kapható szabványos elemekből építkezik, és csak kevés saját elem van a gépeikben — ha egyáltalán van. Különösen biztatóak a többprocesszoros rendszerek; ezek olcsó, ipari szabványnak tekinthető processzorokból, szabványos operációs rendszerrel a nagygépekét elérő vagy meghaladó teljesítményt nyújtanak.

Magyarországon a számítástechnika most lép igazán intenzív fejlődési szakaszba. Jó lenne, ha elkerülnénk azokat a csapdákat, amelyekbe az előttünk járók már beleestek. Most csak körvonalazva a terepet, három nagy veszélyforrás kiküszöbölése jelentene biztonságos helyzetet a jövőre nézve:

1. PC-hálózatok használata olyan feladatokra, amelyekre azok nem képesek.
2. A gyártóspecifikus rendszerek preferálása.
3. Biztos alapok (adatbázisok, információs rendszerek) nélkül drága grafikus megjelenítő rendszerek erőltetése.

Polló László

(A fenti írás második részét lásd ADATRENDEZŐ rovatunkban a 35-36. oldalon.)

„Soft” vagy „hard”?

A szoftveres jövedelmek adója

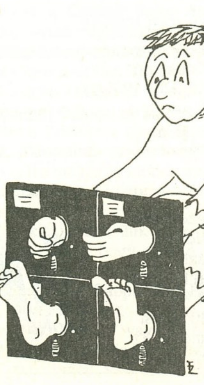
A kérdés: ki lehet-e, illetve ki szabad-e vonni a szoftveres jövedelmeket az adózás alól? A válasz szerintünk az, hogy alapvetően nem. De a kérdésnek több oldala van, nem lehet ilyen egyszerűen sem feltenni, sem megválaszolni. Nézzük itt meg a sok szempont közül az emberi oldalt: mit jelent szoftveresnek lenni, és kik ma Magyarországon a szoftveresek.

Nemzeti-kulturális örökségünk egyik érdekes vetülete, hogy a reál értelmiséget — ezen belül a műszakiakat és a természettudományokkal foglalkozókat — hagyományosan mindig nagyobb megbecsülés övezte, mint akár a humán hivatások képviselőit vagy az üzleti élet, a gazdaság irányítóit. A magyar arisztokrácia még a múlt században is megvetéssel kezelte a kereskedőket. Fokozódott a műszaki értelmiség relatív kiemelkedése a második világháborút követően napjainkig, noha nem a megbecsültség további abszolút növekedésével, hanem a többiek respektjének drasztikus csökkenésével.

Az értékeltolódás talán nem kis mértékben annak köszönhető, hogy a reál értelmiségnek adatott meg egyedül a lehetőség hivatását politikamentesen, politikai értelemben elvileg megalkuvásmentesen gyakorolni. Számtalan tehetséges fiatalembert sodort ez a megfontolás a műszaki, természettudományos pályák felé, pedig más területen talán értékesebbet produkálhattak volna. Kialakult tehát egy színvonalas műszaki értelmiségi gárda — ha úgy tetszik, az általános kontraszelekció egyetlen pozitívumaként.

És ennek a folyamatnak a kellős közepén, a hetvenes években jön a század nagy technikai forradalma — amelyet az átpolitizált társadalomvezetés természetesen nem hajlandó észrevenni —, a számítástechnika robbanásszerű elterjedése.

A jól felkészült, nagy létszámú műszaki értelmiség startképes, kreatív — általában fiatal — rétege azonnal reagál, és fölkapaszkodik az induló vonatra. Autodidakták, nemzetközi segítséggel létesített oktatási intézményekben tanulók, magas szintű külföldi kiképzésben részesülő szakemberek, egyetemi tanszékeken a programozást és a nagy rendszereket csodáló és ismerő, ezek bűvöletében élő specialisták köréből a műszaki értelmiség elitje körvonalazódik — ám a gazdaságnak nincs szüksége a kiművelt emberfők ezen sokaságára.



Nos, ekkor valami sohanem-volt játszódik le: nem az állástalan diplomások számítógépes tagozata alakul meg, hanem a lassanként liberalizálódó gazdasági terepen egyszeriben kiderül, hogy ez a fajta tudás a gazdaság más termékeivel és nemzeti fizetőeszközünkkel ellentétben konvertibilis. Ha pedig eladható, akkor el is adják. Van, akiért 80-100 márkát vagy még többet is fizetnek egy órára. Többet keres, mint honfitársai egy nap alatt — leszámítva az ügyeskedők szintén gyarapodó táborát.

Ezen a ponton jutottunk el az adózásig. Adót minden jövedelem után kell fizetni, ez természetes, ezt többé-kevésbé egyetemlegesen vallja ma az a polgári etika, amelynek jegyében többek között Európához is akarunk tartozni. A kérdés csak az, mennyit. A mai magyarországi adómértékek nem felelnek meg teljesen a fenti követelményeknek — gondoljunk itt a bruttó elvonás százalékaira, valamint arra az európainak nem nevezhető megoldásra, hogy a létminimumra is adót vetnek ki.

Méltánytalan teherrel magától értetődően nem lehet sújtani semmiféle olyan tevékenységet, amelyet szabadon el lehet adni máshol.

Magyarán: ha érvényesül a ma legális adókedvezményeket megszüntetni kívánók akarata, aligha várható tovább, hogy e teljesítmények alanyai akár csak annyi adót is fizetnek, mint eddig. Külföldön fognak adózni, mert ott kevesebbet fizetnek. Az adótáblákat kitervelőknek el kell dönteniük, érdemes-e ezt erőltetni; hoz-e a „szüntessük meg a kedvezményeket” széles körben preferált demagógiája annyit a pénzügy népszerűségi indexén, mint amennyit elvisz az adóbevételekből. Hacsak azt nem kalkulálják bele, hogy egyes nyugat-európai kormányok hálásak lesznek a náluk lecsapódó potya adóbevételekért...

A jelenlegi kedvezményes feltételek mellett is nem kevés szakember vállalja a külföldi adózást, mert így részesülhet az ottani társadalmi juttatásokból (egészségügyi ellátás, nyugdíj, a családtagok perspektivikus érdekei). Mi történik, ha az itthoni lehetőségeket tovább szigorítják?

Minden társadalom célja, hogy a húzó ágazatok produktumát fokozza. Figyelembe kell venni, hogy ma ezt a szerepet a számítástechnika tölti be — ha engedik neki. Ebben a műfajban nemzetközi mércével is kiemelkedő alkotások születnek, amelyeket így is díjaznak. Ennek egyik formája, hogy adókedvezményeket építenek be a jövedelmek elszámolásába.

Bojár Gábor — Hornung Péter

Szakirodalmi válogatás a szerzői jogvédelem témaköréből

Angol nyelvű cikkek:

Software comes clean (A szoftverek jogi védelmének növekvő szerepe)
Electronics (US) 1990/8

Computer law, informatics law or information law? (Javaslatok az információtechnológiával összefüggő kérdések jogi szabályozására)
Information Age (GB) 1990/4

Copyright protection for the status screen of a computer program (Számítógépes programok képernyő-képeinek szerzői jogvédelme)
Rutgers Computer and Technology Law Journal (US) 1989/1

Copyrightability of computer-created works (A számítógéppel létrehozott alkotások szerzői jogvédelmének lehetőségei)
Rutgers Computer and Technology Law Journal (US) 1989/1

Access to federally funded research data under the freedom of information act (A szövetségi kutatási adatokhoz való hozzáférés az információs törvény égisze alatt)
Rutgers Computer and Technology Law Journal (US) 1989/1

Copyright protection: computer-related dependent works (A szerzői jogvédelem kérdései számítógépes adatok és programok részleges, vagy másodlagos felhasználása, módosítása és utánzása esetén)
Rutgers Computer and Technology Law Journal (US) 1989/2

Crackdown a software pirates (Az illegális „szoftvertolvaj" hálózatok ténykedésének veszélyei)
Computerworld (US) 1990/6

Making pirates walk the plank (A szoftverek jogtalan másolása elleni küzdelem)
Computerworld (US) 1990/14

Archival backup copying of software: how broad a right? (Biztonsági szoftvermásolás: mi fér bele?)
Rutgers Computer and Technology Law Journal (US) 1988/2

Defining the scope of software copyright protection for maximum public benefit (A szoftver szerzői jogának meghatározása a maximális közjó érdekében)
Rutgers Computer and Technology Law Journal (US) 1988/1

Software firms unite to duel piracy (Hat nagy szoftverház közös vállalkozása a szoftver-kalózkodás letörésére)
Computer Age (US) 1988/10

Copyright, licensing agreements and gateways (A szerzői jog, a licenc-megállapodások és az adatbázisok információtartalmának jogi védelme)
Online Review (GB) 1990/4

Microsoft tests Rainbow Technologies' antipiracy device (A Microsoft teszteli a Rainbow szoftvervédelmi eszközeit)
Europa Report (GB) 1988/34

Adobe wins copyright for font program... (Az Adobe kapott elsőként szerzői jogvédelmet betűprogramjára)
Computer Age (US) 1990/4

Licencing multiuser software varies by vendor (A többfelhasználós szoftverek licence gyártónként különbözik)
Infoworld (US) 1990/4

You can overcome the fear of shareware (A shareware-rel szembeni idegenkedés leküzdhető)
Computerworld Australia (AU) 1990/33

EEC directive spurs patent controversy (Az Európai Közösség szabadalomvédelmi koncepciója)
Computer Design (US) 1990/6

European Community urged to strengthen protection for US software programs (Az Európai Közösség az amerikai szoftverek védelmének szigorítását sürgeti)
Computer Age (US) 1990/20

The computer and personal privacy, part II: The emerging worldwide response to the threat to privacy from computer databases (A számítógép és a személyiségi jogok: nemzetközi reagálás a számítógépes adatbázisokból fenyegető veszélyre)
Library Hitech (US) 1988/1

Legal aspects of the protection of data centres (A számítóközpontok védelmének jogi vonatkozásai)
Information Age (GB) 1989/2

The data bandits (Az adatbanditák)
Byte (US) 1989/1

Software copying policies of the Fortune 500 (Az illegális szoftvermásolás kérdései az 500 legnagyobb amerikai vállalatnál)
Journal of Systems Management (US) 1989/7

Effect of transborder data flow upon information securty and integrity (A nemzetközi adatáramlás hatása az információ biztonságára és az adatvédelemre)
Information Age (GB) 1989/3

The semiconductor chip protection act and its impact on the international protection of chip designs (Az amerikai félvezető-technológia védelmét kimondó törvény és annak hatása a chipkonstrukciók nemzetközi védelmére)
Rutgers Computer and Technology Law Journal (US) 1989/2

Malpractice in IS? (Számítógépek és informatikai rendszerek hibás működésével kapcsolatos szakmai és jogi felelősség kérdései a joggyakorlatban)
Datamation (GB) 1989/20

To Russia with software (A nyugati szoftvertermékek jogi védelmével kapcsolatos elvi megállapodás szovjet és amerikai szakemberek között)
Datamation (GB) 1989/23

Soup or art? (A szoftvervédelem kérdései és a felhasználó érdeke — szerzői jog, szabadalmi jog, szabvány)
Computer Language (US) 1989/9

Software licencing (A szoftvervédelem jogi kérdései)
Computer Systems Science and Engineering (GB) 1989/3

Data and software security (Adat- és szoftverbiztonság)
Computer Systems Science and Engineering (GB) 1989/6

Storing text retrieval systems on CD-ROM: compression and encryption considerations (Szöveges információkereső rendszerek tárolása CD-ROM-on — az adattömörítés és a kódolás szempontjai)
ACM Transaction on Information Systems (US) 1989/3

Német nyelvű cikkek:

Daten-Desaster verhindern! (Hogyan készítsünk gyorsan és hatékonyan biztonsági másolatokat a merevlemezes tárolóról?)
PC Welt (DE) 1990/12

„Dongles": Hardware schützt Software (A „Dongles" hardveres szoftvervédelem)
Elektronik (DE) 1990/10

Software schützt Software (Szoftveres szoftvervédelem)
Elektronik (DE) 1990/17

Eine harte Nuss für „Knack" (Szoftveres titkosító eljárások)
MC Mikrocomputer-Zeitschrift (DE) 1990/9

Az összeállítás a Sandokan adatbázis alapján készült.
InfoNet Kft.
1111 Budapest
XI., Vahot u. 6.
Telefon: 166-9065

PC Tools 7.0

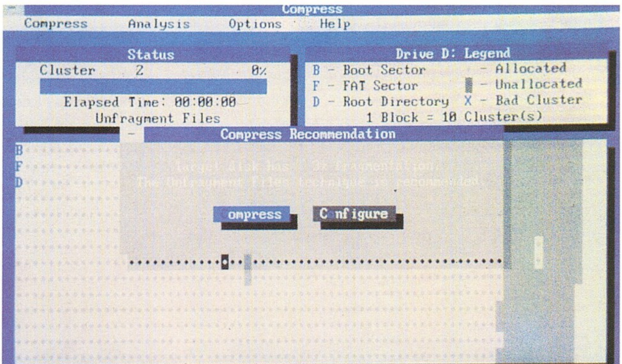
A feneketlen láda

Sokunk első PC-s élménye
és igazi univerzális keretprogramja
a PC Tools valamelyik korábbi változata volt.
Akkor persze még elfért
egyetlen 360 kilobájt kapacitású floppyn.
Júniusban jelent meg
a Central Point PC Tools 7.0 verziószámú változata,
mely már alig emlékeztet
a hajdanvolt ügyes kis PCTOOLS-ra
(vagy ahogy sokan átnevezték: PCT-re),
nagyobb és többet tud, mint valaha.
Ezúttal megvizsgáljuk, mit tartalmaz a Central Point
szinte feneketlen szerszámosládája.
Előrebocsátjuk, hogy rozsdás szögbe nem nyúltunk!

Csak néhány tájékoztató adat a program méreteiről: a PC Tools 14 darab 360 k-s és 1 darab 1,2 Mbájtos lemezen található. A csomagban 8 vastagabb-vékonyabb kézikönyv sorakozik. (Getting Started/Tips for Windows, Commute, Data Recovery and System Utilities, Desktop Manager, DOS Shell/File Manager, Hard Disk Backup, Windows Utilities, Virus Detection, Removal and Prevention.) Valójában öt önállóan is forgalmazott terméket foglal magában a PC Tools 7.0 csomag: a DOS Shellt, a Desktop Managert, a Central Point Backup 7.0-ás változatának külön DOS- és Windows-változatát, a Commute nevű kommunikációs szoftvert, valamint a Central Point Anti-Virus 1.0-át. Ezek egyenként több mint 10 000 Ft-ért kerülnének forgalomba, így egyben azonban 22 000 Ft-ért elérhető a tejles kollekció. A legtöbb programra már hálózatos környezetben is tökéletes működést garantál a gyártó, ehhez a Supervisornak (rendszergazdának) kell a megadott módon installálnia a programokat a központi gép erőforrásaira.

Egy kicsit időzzünk el a mellett a téma mellett is, hogy milyen PC-konfigurációt igényel a PC Tools (a továbbiakban mi is a frappánsabb PCT elnevezést használjuk). Tetszés szerinti olyan PC, XT, AT vagy PS/2, illetve ezekkel kompatibilis gépen futtatható a PCT, amelynek van legalább 512 k memóriája, egy floppy- és egy hard-diszk-meghajtója, s rajta 3.2-es vagy annál újabb DOS üzemel (természetesen 100%-ig kompatibilis az új MS-DOS 5.0-val, ezt jelen sorok írója immár 2 hónapja folyamatosan teszteli). A Commute programhoz két gépet összeköttetésbe kell hozni akár LAN, modem vagy null modem (háromeres soros vonali összeköttetés elterjedt megnevezése) segítségével. Nem árt tehát, ha Hayes-kompatibilis modemmel és faxkártyával is rendelkezünk, így a PCT legújabb szenzációs szolgáltatásait is ki tudjuk használni. Egér megléte nem feltétlen szükséges, de jó tudni, hogy immár minden PCT program egérrel is vezérelhető, és tekintettel a kiválasztható funkciók nagy számára, meg kell vallani, hogy egérrel gyorsabban célt érünk, mint ha a hagyományos billentyűparancsokat adnánk ki. Ha valaki a Windows alá készített szolgáltatásokat is igényli, úgy rendelkeznie kell egy Microsoft Windows 3.x-szel és egy 1,2 Mbájtos floppyegységgel.

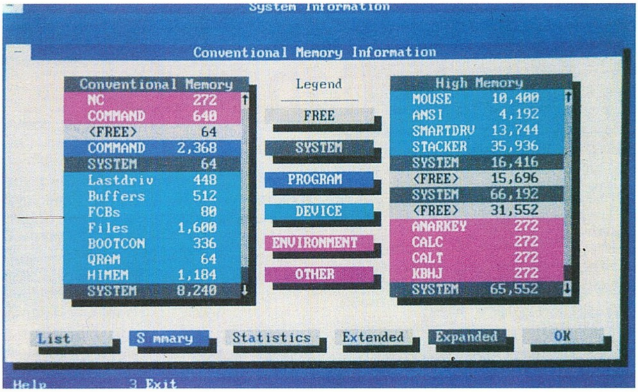
Néhány szó az installációs eljárásról. Itt is számítanak a gyártók arra, hogy valaki pont abból a célból választotta a PCT-t, hogy elveszett(nek hitt) adatait visszanyerje, így mód van azonnal az adat-visszaállító szolgáltatásokat (DISKFIX, UNFORMAT, UNDEL, FILEFIX) is igénybe venni még a merevlemezes installáció előtt (hogy ne tegyük magával az installációval lehetetlenné az elveszett adatok visszanyerését). Nem árt az eredeti lemezekről a DOS DISKCOPY parancsával egy-egy másolatot készíteni, plusz előre legyártani egy rendszerlemezt (FORMAT /S paranccsal). Ez utóbbi lemezre teszi ki a PCT az installáció során — külön kérésre — a legfontosabb rendszerinformációkat (boot-sector, FAT, directory, CMOS RAM tartalma stb.), s a lemezt elkereszteli Recovery Disknek (egy impozáns floppycímkét is mellékeltek a csomagban erre a célra). Vírusfertőzés, rendszerösszeomlás, eláll í-



tódott CMOS Setup, harddiszk-meghibásodás esetén ezernyi gondtól — és szervizestől — kímél meg bennünket ez a lemez, ne sajnáljuk rá a lemezt és időt, fogadjuk meg a gyártók jó tanácsát, csináltassunk egy Recovery Disket!

Az installációs eljárás tovább okosodott, immár szuperintelligens, informatív programmá nőtte ki magát. Kiválaszthatjuk egy menüből, mely részeit kérjük a PCT-nek, melyeket nem, kijelzi a program, hogy a választott opciók mekkora helyet foglalnak majd el a harddiszken, és azon éppen mekkora a szabad hely nagysága. Ha mindent kérünk — mert legtöbben telhetetlenek vagyunk —, úgy 11 Mbájtunk fogja bánni megalomániás döntésünket. A fizikai másolás végeztével kézzel konfigurálhatjuk a felmásolt PCT egyes tulajdonságait: megadhatjuk, mely programok töltődjenek be automatikusan (és tárrezidensen) már a gép újbóli elindításakor is (érdekesség a DeskConnect soros vonali adatátviteli program).

A Desktopon belül kérhetjük, hogy a Backtalk és Fax support funkció dolgozzék-e háttéridőben is (vehessünk, illetve adhassunk fájlokat és leveleket). A Commute is betölthető rezidensen, ekkor lehetőségünk nyílik más gépek erőforrásait is vezérelni, azaz saját gépünkről elindíthatunk egy másik gép harddiszkjén egy tetszőleges programot. Azonnal biztonsági mentést is készíthetünk harddiszkjeink kulcsfontosságú információiról (Mirror), felvértezhetjük magunkat a vírustámadások ellen (VDefend) és beindíthatjuk a véletlen törlés elleni automatikus védelmet (Data Monitor). Ha ezt kérjük, úgy törölt állományok visszaállításakor még az első karakterét sem kell a törölt állománynak kitalálni, azt egy listából egyszerű rámutatással újjáéleszthetjük. Kérhetjük a képernyőkímélő Screen Blanker szolgáltatást, a könyvtártitkosítást (Directory Lock), az általános felülírás elleni védelmet (Write Protection) és a lemezhozzáférést



drasztikusan felgyorsító PC-Cache programot is. Egy ütemező (Scheduler) segítségével tetszés szerinti program (adatmentés, adatkommunikáció) elindítható — például a munkaidő vége előtt 10 perccel.

Háromféle felhasználói szint közül választhatunk — mint a klasszkus nyelviskolában: kezdő, középhaladó és haladó — ennek megfelelően a menük egyre bonyolultabbak és szövevényesebbek lesznek, bizonyos funkciók alacsonyabb szinten (különösen a visszafordíthatatlan változtatásokat is okozó parancsok) el sem indíthatók. Megszabhatunk egy jelszót is a PCT-hez, így annak szolgáltatásait csak az veheti igénybe, aki a jelszót ismeri. Egy szemléletes ábra megjeleníti, hogy mennyi memória áll még szabadon rendelkezésünkre a konvencionális 640 k-ból. Kérésre a PCT az AUTOEXEC.BAT és CONFIG.SYS állományainkat is kitölti a megfelelő módon — de még előtte egy-egy biztonsági mentést készít *.SAV kiterjesztéssel.

A PCT helprendszerét alapjaiban dolgozták át, immár egy keresztreferenciákkal ellátott hipertext rendszernek tekinthető önálló elektronikus kézkönyvként funkcionál (névmutató-index is van benne).

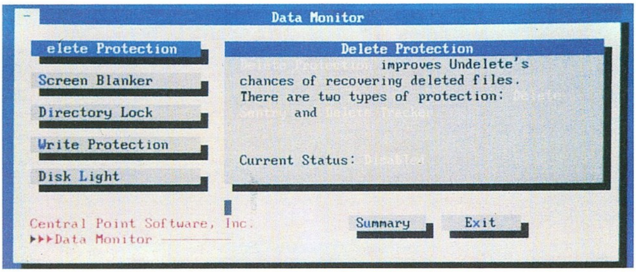
EGA vagy VGA monitoros gépen már karakteres üzemmódban is nyíl alakú egérkurzorunk lesz, és egyértelműen az ikonszervezésű, rámutatásos kezelői felület felé tett nagy lépést a Central Point. Vízszintes és függőleges scroll-bar segíti az egérrel történő gyors pozicionálást. Azt kell mondanunk, hogy a PCT-t formatervezők és plasztikai sebészek vették kezelésbe, s az átalakítás csinosabbá, elegánsabbá, áttekinthetőbbé tette a jó öreg, de még nem ráncos PCT-t. Akinek a gyári színek, menük nem tetszenek, azok természetesen a szivárvány összes színét felhasználva átfesthetik a PCT-t a PCCONFIG segítségével.

A windows-os szolgáltatásokról dióhéjban: minden egyes DOS-program számára is ízléses ikon szolgál, ezeket DOS-ablakban futtatja a Windows. Négy speciális valódi Windows-felhasználást is kapunk: a CP Backupot, a CP Schedulert, a CP Launchert (ez a sokoldalú és gyors program a Windows File Managerét helyettesíti nagy sikerrel) és az Undelete-et. Néhány program nem indítható el a Microsoft Windowsból, sőt tilos is elindítani, ilyen például a COMPRESS.

Különös tekintettel az újdonságokra, néhány mondattal áttekintjük a PCT legfontosabb részeit, programjait.

CP Commute

Távoli terminált vezérlő (remote controlling) program. Főbb alkalmazási területei: otthon dolgozók postai telefonvonalon adhatják át munkáikat megbízójuknak, több gépről is lehet ugyanazon a projekten, táblázaton, levélen dolgozni, mindenki hozzáteheti a ma-



gáét. Végfelhasználókat kisegíthetjük akár több száz kilométernyi távolságból is az általuk elindított problémás program monitorozásával (átvehetjük a távoli billentyűzet vezérlését, az ottani képernyőtartalom látszik a mi monitorunkon is!).

Hivatásszerű távoktatásra is alkalmas!

CP Backup for Windows

Ami merőben új: menthetjük a harddiszk-tartalmat, míg egy másik programon is dolgozunk (valódi multitasking). Egyszerre több merevlemezt is kezelhetünk. Szabványos streamereket kezel, kompatibilis a DOS alatti CP Backuppal, a mentés során automatikus vírusdetektálással kiszűri a fertőzött állományokat. Nagyon jól használható a Compare Data opció, amikor is egy mentett állomány és az aktuális harddiszk-tartalom kerül górcső alá. A DOS alatti CP Backup 7.0 több mint 30-féle adatformátum natív megjelenítésére is képes, új, ún. Express menüfelületet kapott a minél egyszerűbb kezelés érdekében.

PC Shell

Változtatható méretű, eltüntethető ablakok, szemléletes fájlszimbólumok, ikonok, új menüfelület (de kérhető a régi v6.0-ás is), jelszóval is védhető. DOS parancssor, az utolsó 16 DOS-parancs visszahívható egy ablakba. Saját, felhasználói (User Menu) is létrehozható. Laptop gépek összekötésekor szoftverkapcsolatot létesít, gyors fájlátvitelt produkál.

A Desktop nem újdonság, mégis érdemes átfutni, milyen szolgáltatásokat kínál: többablakos editor, angol nyelvű helyesírás-ellenőrzővel, vázlatkészítő (outliner), interaktív dBASE-kompatibilis adatbázis-kezelő, automatikus telefontárcsázó, határidőnapló prioritásos tennivaló-listával, szabad időpont keresésével, modemkommunikáció (MCI Mail is!), elektronikus postázó, faxkártyakezelő, makróeditor scriptnyelvvel, négyféle kalkulátor (algebrai, üzleti, tudományos és programozói), ASCII táblázat stb...

Diskfix

Bizonytalan floppyt újraformáz adatvesztés nélkül, merevlemezen karbantartó jelleggel nem destruktív alacsony szintű formázást hajt végre, harddiszket optimalizál interleave faktor szempontjából, felületellenőrzést végez a várható meghibásodások előjelzése céljából, rejtett, lappangó vírusokat derít fel a rendszerterületen belül.

FileFix

Sérült Lotus 1-2-3- (WKS, WK1-től WK3-ig) és Symphony-, valamint dBASE-kompatibilis állományok kijavítása. Utóbbiaknál elbánik a sérült headerrel rendelkező fájlokkal, a ZAP-polt DBF-ekkel, a record frame errorral és az állomány közbeni véletlen EOF jellel, valamint az illegális karakterekkel.

PCSecure

Adatfájlok titkosítása jelszóval (melyeken egyébként tetszés szerinti DOS-művelet elvégeztethető), tömörítés, kódolt állományok adása, visszafejtése modemen keresztül.

SI, System Information

Részletekbe menően meghatározza az aktuális hardverkonfigurációt, a memóriakiosztást, a bővítések típusát, méretét, sebességét. Winchester- és CPU-tesztet készít. Hálózati környezetben kiírja az egyes felhasználók, volumenok, felhasználói csoportok nevét, jellemző adatait, az SI rövid üzenetek küldésére is alkalmas a hálózaton keresztül (Vol Info, User List, Group List, File Info, Details, Select User, Message stb.)!

Az egyik legjobban használható apró parancssori segédprogram az MI.COM (Memory Info), mely részletesen megmutatja, milyen állományok, eszközmeghajtók (device drivers) csücsülnek a memóriában 640 k felett és alatt, valamint a memóriabővítésekben. Nélkülözhetetlen információkat ad! Az új FileFind program már kérésre felderíti az azonos nevű és azonos tartalmú állományokat is, hogy aztán megtizedelhessük őket. A különböző adatformátum-megjelenítők nemcsak a PC Shellből érhetők el, automatikusan használhatók az Undelete, FileFix, FileFind és CP Backup programokból is. A fontosabb viewerek: dBASE, MS Works, Paradox, R:BASE, Lotus, Symphony, Excel, Multiplan, Quattro (Pro), MS Word, WordPerfect, WordStar, XyWrite, .ARC, bináris, LHARC, .PAK, .PCX, .ZIP, .ZOO.

Directory Maintenance

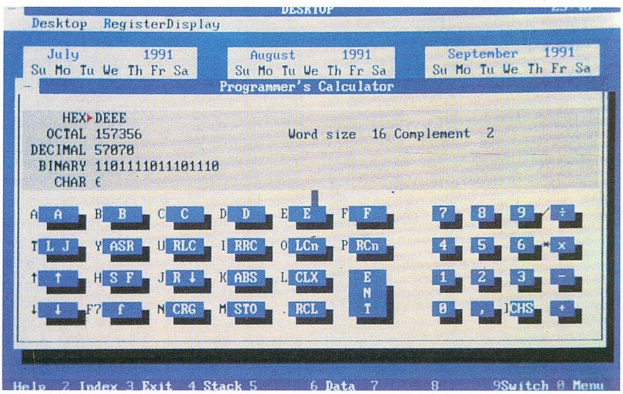
Egy zseniális újdonság, a Norton NCD alapján készült, de annál szemléletesebb és gyorsabb! Kiírja az egyes könyvtárak méretét is, azt kis oszlopokkal szemlélteti. Teljes könyvtárstruktúrák másolására, áthelyezésére, törlésére is képes. Képes újraszervezni a könyvtárstruktúra rendezettségét.

A PC Shell és PC Desktop automatikusan feltölti kódjának tetemes részét a megtalált EMS vagy extended memóriabővítésbe. Ha ilyennel nem rendelkezünk, úgy a harddiszkre menti átmeneti állományait a PCT, így persze egy kissé lomhább lesz, de alig foglal el több mint 12 k-t a 640-ből (lásd SWAPPS, SWAPDT).

A Central Point Anti-Virus 1.0-ról — noha része a csomagnak — ezúttal nem írunk, legközelebb részletes teszttel szolgálunk róla összevetve egyik legnagyobb konkurensével, a Norton Anti-Virus 1.5-tel.

Zárszóként csak annyit, aki teheti, vegye meg a programot, rövid időn belül többszörösen is meg fogja szolgálni az árát. Szakember szerszám nélkül, számítástechnikus PC Tools nélkül olyan, mint a versenyló láb nélkül a kitakarítatlan istállóban.

Herczeg József



Shooting Gallery

Veszélytelen lövöldözés

Bizonyosan mindenkit megérintett már a vásári céllövöldék varázsa. A versenyszellem pedig mindannyiunkban benne van. Ezért valószínűleg széles körben tetszést arat ez a számítógépes játék, amely nemcsak kellemes vadászat lehetőségét kínálja, hanem grafikájával is gyönyörködtet.

A játék hat menetből áll. Az egyes menetekben mások és mások a célpontok, feladatunk azonban nem változik: olyan gyorsan kell lőnünk a célpontra, ahogyan csak bírunk.

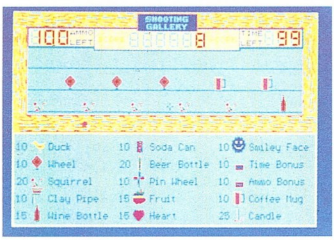
Egy játék során 100 másodperc alatt 100 lövést adhatunk le a három sorban különböző sebességgel megjelenő figurákra. Időnként jutalmat is kapunk, vagy plusz idő, vagy plusz töltény formájában.

Az egyes találatokért különböző pontszám jár, attól függően, hogy mit és melyik sorban találtunk el. A gyorsabban mozgó sorokban elért találatok több pontot érnek.

Bármikor abbahagyhatjuk a játékot az ESC billentyű leütésével. Az 1-es és a 4-es menetben szünetet is tarthatunk, ehhez a space-t kell leütnünk. A többi menetben már automatikusan benne vannak a szünetek.

1. menet

Egy igazi céllövöldében vagyunk. Az egyes tárgyak más-más pontszámot érnek, ezt a képernyő alsó sorában láthatjuk. Plusz pontokat kapunk, ha a gyorsabb sorokban találunk el valamit: 5 pontot a középső sorért, 10 pontot a leggyorsabb, felső sorért.



Különleges boxok is megjelennek, ezeket is érdemes lelőnünk. Leghatékonyabban úgy növelhetjük pontszámunkat, ha egy teljes sor összes figuráját eltaláljuk. A soronkénti tárgyak száma 4 és 8 közötti, 100 vagy akár 250 jutalompontot is összegyűjthetünk.

Tárgyak	Számuk soronként	Jutalompontok
Kacsa	8	100
Kerék	8	100
Mókus	6	100
Agyagpipa	8	100
Borosüveg	6	100
Dobozos üdítő	8	100
Sörösüveg	4	200
Kerék	6	100
Gyümölcs	6	100
Szív	6	100
Smiley (nevető fej)	8	100
Kávéscsésze	6	100
Gyertya	4	250

2. menet

Itt agyaggalamb-lövészetben jeleskedhetünk. A felröppenő célpont gyorsan halad, távolodva egyre kisebbé válik. A képernyő alján látjuk a figyelmeztetést: kissé a galamb elé kell céloznunk. Időbe telik ugyanis, hogy a lövedék elérje a célpontot. Téves lövésért természetesen nem kapunk pontot. Találat esetén a pontok száma attól is függ, hogy milyen gyorsan találtuk el a galambot.

3. menet

Céltábla közepébe kell az egérrel beállnunk, miután egy közlekedési lámpa zöldre váltott. Gyorsan rá kell kattintanunk, mielőtt ismét piros lenne a lámpa. A kapott pontszám nemcsak attól függ, hogy milyen gyorsak voltunk, hanem a pontosságtól is. Lehetőleg a céltábla középső, piros részére álljunk rá lövés előtt!

4. menet

Hasonló az 1. menethez, de kissé bonyolultabb. Mások a tárgyak, amikre lövünk, a pontszámok pedig megduplázódtak. Nemcsak egy irányba mozognak a figurák, időnként visszafordulnak. Jön egy-egy „HIT ME" feliratú célpont is, ezt tényleg próbáljuk meg lelőni, máskülönben 500 pont levonása a büntetésünk.

Az 1. menethez hasonlóan itt is az a legjobb, ha teljes sorokat tudunk kilőni. Itt a soronkénti tárgyak száma 4 és 8 közötti, a jutalompontok pedig 200 és 500 között lehetnek.

5. menet

A 2. menet nehezített változata. Kétlövetű puskánkkal két galambra lőhetünk.

6. menet

Hasonló a 3. menethez. Itt is a zöld jelzés alatt lőhetünk, csak most két céltáblára, két tölténnyel. Az 5. menetből már kikövetkeztethető, hogy itt is minden találat számít, nem szükséges feltétlenül a dupla találat, az időnkbe azonban beszámít a sikertelen lövés is.

Mielőtt belevetnénk magunkat a játékba, a kezdő képernyőről válasszuk ki a gyakorlás opciót! Nézzük meg, ki milyen pontszámot ért el eddig! Sokkal érdekesebb lesz így a verseny, ha tudjuk, hány pontot kell elérnünk ahhoz, hogy mi legyünk a legjobbak.

A játék grafikája kiemelkedően jól sikerült. Biztos a siker: nemcsak a gyerekek fognak lelkesen lövöldözni, hanem valószínűleg a szülők is versenyre kelnek csemetéikkel. VGA monitor és egér szükséges a játékhoz.

A Shooting Gallery a Solarsoft programkönyvtár #350 sorszámú lemezén található, 4 másik VGA játékkal együtt. (Ezek közül a Solitile nevű, MahJongg jellegű játékprogram is feltétlen sikerre számíthat.)

Verebély Pálné

Vonalhúzás és körhúzás

Kevesen kerülnek olyan helyzetbe, hogy a vonalhúzó vagy a körhúzó rutinokat maguknak kelljen megírni. Ez szinte minden fordítóprogram könyvtárában megtalálható. De tényleg: Hogyan is működnek ezek a rutinok?

Kedves Olvasó, mielőtt legyintene és továbblapozna, válaszoljon őszintén: meg tudna Ön írni egy hatékony vonalhúzó, vagy körhúzó rutint? Egy olyat, amely összemérhető sebességű a Turbo-Pascal vagy a C++ könyvtári rutinjával?

Ha úgy érzi, hogy igen, próbálja meg, mielőtt tovább olvasna. Hogy gyors legyen, a következőket kell betartania:

1. Szorzás, osztás és lebegőpontos művelet használata tilos.
2. Egy pontot csak egyszer szabad rajzolni.
3. A vonalnak folytonosnak és egyenletesnek kell lennie.

A probléma megoldásának látszólag legegyszerűbb módja, hogy keresünk egy okos könyvet, amelyben benne van az algoritmus. Sajnos a legtöbb ilyen könyvet matematikusok írták más matematikusoknak, ezért az átlagprogramozó számára csaknem érthetetlenek. Programot a legritkább esetben találunk, és azt is Algolban, vagy valamilyen más, sose hallott programnyelven. Így hát jobb, ha kitaláljuk az algoritmust magunk.

Vonalrajzolás

A vonalhúzó algoritmushoz többféleképpen eljuthatunk most. Itt három eltérő megközelítést mutatunk be. Segítséget jelent, hogy csak a (0,0) pontból az (x1,y1) pontba menő egyenes rajzolását kell kitalálnunk, mivel ebből eltolással, a tengelyekre tükrözéssel, és 90 fokos forgatásokkal bármely egyenes előállítható. Sőt, még azt is feltehetjük, hogy x1>=y1.

1. Az egyenletből kiindulva

Már az általános iskolában tanítják, hogy a (0,0) ponton, és az (x1,y1) ponton átmenő egyenes egyenlete
y=(y1/x1)*x

Ha kihasználjuk, hogy x irányba mindig csak egyet kell lépni, az 1. algoritmushoz jutunk. Ennek hibája, hogy y-nak valós változónak kell lennie.

Válasszuk szét y egész (inty) és törtrészét (fray). Ekkor jutunk a 2. algoritmushoz. Ez még semmivel sem jobb, mint az első, mivel a fray-nak még mindig valós változónak kell lennie.

Legyen fray helyett x1fray=x1*fray! Módosítsuk az algoritmust mindenütt, ahol a fray előfordult. Ekkor kaphatjuk meg a 3. algoritmust. Ebben már mindegyik változó lehet egész, szorzást nem tartalmaz, tehát a fent ismertetett követelményeknek megfelel.

(A példák azért nem használják a C-ben megszokott, és a programban is szereplő rövidített jelöléseket, hogy C-ban nem járatos olvasóink számára is érthetőek legyenek.)

2. Grafikus megközelítés

Nagyítsuk fel a raszterhálót x1-szeresére (1. ábra). Tegyük fel, hogy ha a finomított raszterháló egy mezőjére leteszünk egy pontot, akkor az szétterül és kitölti az eredeti háló egy mezőjét. Ekkor csak minden x1-edik pontot kell kirajzolnunk. Ha x1-et lépünk x irányba, akkor y1-et kell y irányba menni.

Használjunk két változót: az egyik tárolja azt, hogy az eredeti raszterháló melyik mezőjén vagyunk, a másik pedig azt, hogy a mező szélétől hány finomított rasztrerpontra. Ha ezeket a változókat inty-nak és x1fray-nak hívnánk, visszakapnánk a 3. algoritmust.

Ez a grafikus megközelítés rámutat az algoritmus kerekítési hibákból származó tökéletlenségére. A mágneslemezen már a javított algoritmus van.

3. Invariáns keresésével

A szorzástól úgy szabadulhatunk meg, hogy bevezetünk egy alkalmas változót, amelynek minden pontban tudjuk az értékét, és segítségével egyszerűen el tudjuk dönteni a következő pont helyét. A matematikusok az ilyen változót invariánsnak hívják. A
dd=y1*x-x1*y
invariáns értéke akkor a legkissebb, ha a vonalon vagyunk. Minthogy bármely P(x,y) pontban csak azt kell eldönteni, hogy a következő pont A(x+1,y) vagy B(x+1,y+1) legyen (l. a 2. ábrát).

Az invariáns értéke:
dd(x,y)=y1*x+x1*y
dd(x+1,y)=y1*(x+1)+x1*y=
dd(x,y)+y1
dd(x+1,y+1)=y1*(x+1)+x1*(y+1)=
dd(x,y)-x1+y1

Látható, hogy az invariáns következő értékét egyszerűen, szorzás nélkül meg tudjuk határozni. A két új érték abszolút értékének összehasonlításával pedig eldönthető, hogy melyik lesz a következő pont.

Ha ez alapján írjuk meg a programot, akkor is a 3. algoritmushoz hasonló eredményre jutunk.

Kör rajzolása

Kör rajzolása egy fokkal már bonyolultabb. Itt is felhasználhatjuk a szimmetriát és feltételezhetjük, hogy a középpont (0,0). A pontokból összeállított kör egy nyolcadrészét kell csak megrajzolni, a többi ennek tükrözésével és forgatásával megkapható.

Az algoritmust a vonalrajzolásnál bemutatott módszerek bármelyikével levezethetjük, legegyszerűbben azonban az invariáns keresésével.

Az invariáns a kör egyenletéből származtatható:
i=x*x+y*y-r*r

Ennek értéke pozitív minden a körön belüli pontra.

A P(x,y) pontban azt kell eldönteni, hogy a következő pont A(x+1,y) vagy B(x+1,y-1). Az invariáns következő értéke így számítható:
i(x,y)=x*x+y*y-r*r
i(x+1,y)=(x+1)*(x+1)+y*y-r*r=
x*x+2*x+1+y*y-r*r=i(x,y)+2*x+1
i(x+1,y-1)=(x+1)*(x+1)+(y-1)*(y-1)-
r*r=x*x+2*x+1+y*y-2*y+1-r*r=
i(x,y)+2*x-2*y+2

Az előjelből eldönthető a következő pont helye. Az algoritmus kerekítési hibák miatti módosításokkal megtalálható a mágneslemezes melléklet DRAW.C állományában.

Aki még további feladatot szeretne, próbálkozzon meg az ellipszis rajzolásával. Mert a Turbo-C++ és a Turbo-Pascal ellipsziseket rajzol, amikor a circle( ) eljárást meghívjuk, azért, hogy kiküszöbölje a képernyő torzítását. Sok sikert! (Segítségül: két invariáns kell hozzá.)

Aki pedig csak játszani akar, hasonlítsa össze ezeknek a rutinoknak a sebességét és működési módját a könyvtári eljárásokéval. Érdekes, hogy a Turbo-C vonalhúzó rutinja nem ugyanazokból a pontokból rajzolja meg a vonalat. Az eltérés olyan kicsi, hogy csak akkor látható, ha az itt ismertetett rutinnal húzott vonalat a könyvtári rutinnal próbáljuk letörölni.

Pintér Gábor

A Scan 7.2C77

Kisebb és többet tud

A McAfee-féle Scan program új verziója kisebb és gyorsabb az Alaplap májusi számában ismertetett 6.8.V7.4 verziónál. Természetesen a vele detektálható jószágok száma növekedett, miközben a program méretét 60 kB alá szorították. A 77-es verzió a szerzők szerint 243 vírust ismer fel, összesen 505 különböző változatban. Ezek ismertetése megtalálható a VIRLIST.TXT állományban.

A program két új kapcsolóval bővült, melyek a batch alkalmazásokat támogatják. Ezek a /NOPAUSE és a /NOBREAK. A Scan amellett, hogy detektálja a vírusokat, arra is használható, hogy a fertőzött állományokat kiirtsa, felülírva azokat hexa C3 értékkel. Ez az eljárás visszahozhatatlanul törli az érintett állományokat. A program a különálló és a hálózati gépeken is fut, de serveren nem. Oda a NETSCAN való. A Scan futtatásához legalább 256 k RAM és DOS 2.00 feletti verzió kell.

A program használata:

SCAN d1:[path][file] ... d10:[path][file] [kapcsolók]

Mint látható, akár tíz meghajtót/partíciót is ellenőriztethetünk egyetlen jól irányzott paranccsal. Ha a meghajtónév után \ jelet teszünk, akkor csak az adott meghajtó gyökérkönyvtárát és a bootszektort ellenőrzi. Ez közvetlenül az adott meghajtónév után írandó, szóköz vagy egyéb elválasztó karakter nélkül. Ha kiterjesztéssel fájlnevet is megadunk (joker karakterek megengedettek a DOS-konvencióknak megfelelően), akkor a Scan csupán a megadott állományokat ellenőrzi.

A kapcsolók

/A

Minden egyes állományt ellenőriz, ezért eléggé megnöveli az ellenőrzés idejét. Két esetben ajánlatos a használata:

— Ha új programokat akarunk ellenőriztetni.
— Ha talált vírust a Scan.

/AV

10 bájt hosszú ellenőrző kódokat ragaszt az adott EXE és COM állományok végére. Ha nemcsak egy könyvtárat ellenőriztetünk vele, hanem teljes meghajtót, akkor a lemez bootszektorához, partíciós táblájához és a rendszerállományokhoz is generál egy ellenőrző összeget, amit a tesztelt lemez gyökérkönyvtárában rejtett állományban helyez el SCANVAL.VAL néven.

/CV

Figyeli az adott állományok ellenőrző kódjait, melyeket az /A kapcsolóval előzőleg hozzáfűztünk. Mintegy 25%-kal lassítja le a programot.

Egyes régebbi típusú Hewlett-Packard és Zenith PC-k rendszerindításkor módosítják a bootszektort vagy a partíciós táblát, amit a SCAN azonnal jelez, amennyiben a /CV kapcsolót használjuk.

/D

A fertőzött állományokat felülírja hexa C3 értékkel, majd törli azokat. Az így kiradírozott állományok nem hozhatók vissza. A bootszektort és a partíciós táblát fertőző gyönyörűségek nem távolíthatók el a lemezről a /D kapcsolóval. Erre a feladatra használjuk a Clean programot vagy egyéb vírusirtót.

/E .XXX .YYY .ZZZ

A megadott kiterjesztésű állományokat is ellenőrzi a szokásos COM, EXE, BIN, OV?, PGM, PIF, PRG, SYS és XTP állományokon kívül. Az /A kapcsoló mellett felesleges, mivel abban minden benne van. Egyszerre legfeljebb három kiterjesztést adhatunk meg. Be kell írni a pontot is, és az egyes kiterjesztéseket szóközzel kell elválasztani.

/EXT d:víruslista fájl

A megadott adatfájlból vett információk alapján keres vírusokat. A víruslista fájlszerkezete és használata megegyezik azzal, amit az Alaplap májusi számában közöltünk.

/FR

Franciául írja ki az üzeneteit.

/M

A memóriát nemcsak a leggyakoribb, hanem az összes vírusra ellenőrzi. Az /M kapcsoló 6-20 másodperccel növeli meg a Scan futásidejét.

/MANY

Több lemezt is tesztelhetünk anélkül, hogy a Scant újra meg újra be kellene olvastatnunk. Ha a rendszer vírusmentes, és csak a lemezeket akarjuk ellenőriztetni, akkor a /MANY kapcsoló a /NOMEM kapcsolóval kombinálva jelentősen meggyorsíthatja az ellenőrzést. Egy-egy lemez ellenőrzése után a SCAN ilyenkor rákérdez, hogy akarunk-e még egy lemezt ellenőrizni.

/NLZ

Az LZEXE segítségével tömörített állományok ellenőrzését elhagyja.

Figyelem!

Egyes .EXE állományokat a Clean nem tud helyreállítani. Ez olyankor fordul elő, ha a fájl belső overlayt tartalmaz, s a vírus nem a fájl végéhez írja saját kódját, hanem annak belsejébe. Ilyenkor ki kell törölni a sérült állományt és az eredeti lemezről újra bemásolni.
A Stoned vírus eltávolítása a nem standard formátumú merevlemezeken a partíciós tábla megrongálódásával is járhat, ami a lemez minden adatának elvesztését jelenti. Ezért célszerű a Clean futtatása előtt a fontosabb adatokat kimenteni.

/NOBREAK

Főleg batch fájlokban hasznos, ha le akarjuk tiltani a megszakítást. A /NOBREAK használatakor sem a ^C, sem a ^BREAK nem szakítja meg a Scan futását.

/NOMEM

Nem ellenőrzi a memóriát. Csak akkor használjuk, ha előzőleg már ellenőriztük rendszerünket és az vírusmentes. Főleg a /MANY kapcsolóval együtt szokás használni floppylemezek ellenőrzésére.

/NOPAUSE

Amikor a(z egyik) céllemezen várhatóan több vírus is található, a futás gyorsítására használjuk a /NOPAUSE kapcsolót.
Ennek hatására a Scan nem áll meg, ha egy-egy képernyőt teleír a talált adatokkal.

/REPORT d:<fertőzött.lst>

A megadott állományba írja ki a fertőzött állományok megfelelő adatait.

/RV

Az ellenőrző kódokat, amit az /AV kapcsolóval előzőleg felrakattunk, leszedi az adott állományokról. Természetesen nem kombinálható az /AV kapcsolóval.
Kilépéskor a Scan az alábbi ERRORLEVEL értékeket állíthatja be:

0 Nem talált vírust.
1 Egy vagy több vírust talált.
2 Szokatlan megszakítás (programhiba).

Ha a felhasználó szakítja meg a programot a ^C vagy a ^BREAK megnyomásával, akkor az ERRORLEVEL 0 vagy 1, attól függően, talált-e a megszakítást megelőzően vírust.

A Clean 7.2V77 használata

A Clean a McAfee cég fertőtlenítő programja, amit a Scannel detektált vírusok irtására használhatunk. A Clean 75 vírust és azok mutációit képes kiirtani.

A szintaxis:

CLEAN d1: ... d10: [víruskód] <kapcsolók>

ahol
a d1: ... d10: a fertőtlenítendő meghajtók nevei (egy menetben legfeljebb tíz), a [víruskód] pedig a Scan által megadott vírusazonosító kód. A [és] jeleket is be kell írni.

/A

A lemezen minden fájlt megtisztít az adott vírustól.

/E .xxx .zzz .yyy

Az adott kiterjesztésű állományokat fertőtleníti.

/FR

Franciául üzen a program.

/MANY

Több lemezt is fertőtlenítünk.

/NOPAUSE

Nem áll meg egy-egy képernyő teleírása után egy-egy billentyűlenyomásra várva.

/REPORT d:<listafile>

A fertőtlenített állományok listáját kiírja a megadott állományba.

Nagy Gábor

Ha a Scan vírust talált

1. Először is írjuk fel, hogy mit talált és hol. A kiírt kódot a megfelelő fertőtlenítő program indításakor használhatjuk.
2. Indítsuk újra a rendszert egy steril, leragasztott rendszerlemezről.
3. A megadott információ és a VIRLIST.TXT alapján kiválasztott fertőtlenítő programmal takarítsuk ki a lemezt. Nem árt kézbe venni az Alaplap Könyvek sorozatban megjelent Víruslélektant, sok hasznos tanácsot találhatunk benne arról, mit hogyan irtsunk, illetve mivel és hogyan ne. (Hamarosan megjelenik az Új Víruslélektan is!)
4. A Scant futtassuk le a fertőtlenített lemezen az /A /M kapcsolókkal. Ha még maradt rajta valami, azt is takarítsuk ki.

SolarSoft sikerlista

Az 1991. májusi és júniusi eladások alapján

	No.	Programnév	Db	Programleírás
1.	421	PKZ110 & PKLITE & SHEZ	1	A sűrítés magasiskolája és Norton Commandere
2.	319	SCAN76 & NETSCAN	1	McAfee-féle vírusmegelőző, -detektor és -ölő
3.	470	MULTI-EDIT 5.0	1	A világon a legjobbnak tartott editor
4.	510	ARJ 2.10	1	A leghatékonyabb adattömörítő
5.	475	NEWSPACE	1	Megduplázza merevlemezünk kapacitását
6.	466	SKYGLOBE STAR GAZER	1	Mozgó csillagtérkép, PC-planetárium
7.	509	ZIPVIEW & LHA 2.10	1	A Norton Commander kiegészítése + az új LHARC
8.	494	TEGLP WINDOWS TOOL	1	Ikongrafikus felület, ikoneditor TP-hez
9.	507	PC-BROWSE & NG	1	NG decompiler és egy szenzációs NG-klón
10.	096	AS-EASY-AS 4.00P	1	Lotus-kompatibilis egyszerű táblázatkezelő
11.	508	PROCUBE LITE 1.0	1	A Qubecalc továbbfejlesztése, valódi 3D
12.	505	GALAXY LITE 1.6	1	A nagysikerű Galaxy szövegszerkesztő felújítása
13.	435	OPTIKS 2.18 & ICONVERT	1	PCX, PIC, GIF, TIF, GEM, MAC... grafikus konverterek
14.	485	BASIC COMPILER	1	Két ragyogó fordító — editorral
15.	463	GAMES FOR WINDOWS	1	10 + 1 játék MS Windows 3.0 alá
16.	477	BACK & FORTH	1	Memóriamenedzser: 20 programot futtat egyszerre
17.	383	4DOS V3.01a	1	COMMAND.COM-pótló DOS-héj: 50 új parancs
18.	474	JORJ POP-UP DICT.	2	58 000 szavas angol értelmező szótár
19.	480	GRASP 1.10C	1	Látványos animáció- és demókészítő program
20.	432	LZEXE & LIST 7.5e	1	Gyors EXE-kompresszor, Vernon Buerg LIST PLUS-a
21.	506	PROLITE 1.01	1	A PC-Write szövegszerkesztő egyszerűbb változata
22.	425	POP-DBF 1.1 & DLITE	1	Tárrezidens dBASE (EDIT/BROWSE/DISP stb.)
23.	511	4EDIT & 4ZIP	1	Hasznos kiegészítések 4DOS-hoz (SolarSoft #383)
24.	304	TURBO TECHNO JOCKS	2	Szuper Turbo Pascal-unitok, forrásaikkal együtt
25.	472	SHARESPELL	1	Bővíthető, önálló helyesírás-korrektor
26.	484	SR-INFO	2	dBASE-kompatibilis fejlesztőrendszer
27.	461	ZEPHYR 2.0	2	Komplett, interaktív adatbázis-kezelő
28.	329	PC-MAGAZINE BENCH.	1	Hardvertesztek szervizeseknek (v5.0)
29.	468	SUPER ASSEMBLER ED.	1	TASM-ra kihegyezett programeditor
30.	478	XTREE 2.0E	1	Kisméretű, gyors fájlmenedzser

Kommentezés

Egy iZIPicit elegánsabban

A PKZIP — hasonlóan a többi felsőosztálybeli tömörítő programhoz — lehetővé teszi egy már használható méretű megjegyzés hozzáfűzését a ZIP állomány egészéhez (= ZIPkomment) és az egyes bepakolt állományokhoz (= komment). Ez a lehetőség — mint ismert — a PKZIP –z és –c parancsai segítségével áll rendelkezésünkre a legegyszerűbben. A kommentező parancsok használata esetén a program a ZIP fájl létrehozásakor, illetve módosításakor bekéri a ZIPkommentet, illetve a kommentet, s a PKZIP 127 karakter hosszúságig hajlandó azt elfogadni.

A kommentezés nagyon hasznos lehetőség, de jó lenne, ha módunkban állna előre megszerkesztett, esetleg ANSI vezérlőkódokat is tartalmazó listaállományból beolvastatni a (ZIP)komment szövegét, és túlléphetnénk a 127 karakteres korlátot.

A SolarSoft programkönyvtár ZIPCOMNT.EXE (#343, ZIP Utilities lemez) programja volt az első, amit megpróbáltam használni. Ez a program már saját leírása szerint is csak 117 karakterig fűzhet ZIPkommentet a ZIP fájlhoz, de maga a program szerzője azt javasolja, hogy ne nagyon írjunk 79 karakternél hosszabb szöveget ide. A program lehetőséget ad arra is, hogy egy előre megszerkesztett komment állományból pakoltassuk át a komment szövegét. Ehhez csupán annyit kell tennünk, hogy definiálunk egy ZIPCOMNT változót a program futtatása előtt. Itt néha problémát okozhat, ha betelt a változóknak fenntartott tárterület. Ilyenkor nem tesz be kommentet a program. Ha az összes ZIP fájlhoz azonos kommentet fűzünk, akkor jó ez a megoldás, ha azonban a kommenteket gyakran váltogatjuk, célszerűbb más megoldást keresni.

Itt állok tehát egy megszerkesztett logóval, amit szeretnék az összes általam kiadott ZIP fájlra feltenni.

Nincs megoldás? A BBS-ekből lehívott állományok jól megszerkesztett, terjedelmes logói azt tanúsítják, hogy van. Közreadom, mire jutottam én.

A NoGate Consulting cég PAK programja (SolarSoft, #431 lemez) azt nyújtja, amit vártunk tőle, egyéb egyedülálló szolgáltatásairól nem is szólva.

Miért a PAK?

— Tetszőlegesen konvertálhatunk a PAK, ARC és ZIP formátumú állományok között oda-vissza, segédprogram nélkül generál PAK-SFX-et.

— A PAK nemcsak a parancssorban fogad el listafájlt, hanem a megjegyzések (fájl-, illetve archív komment) bekérésekor is. A listafájl és így a benne lévő megjegyzés hossza nem korlátozódik egyetlen sorra.

— Viszonylag egyszerűen, a PAKINST segédprogram és kedvenc szövegszerkesztőnk segítségével (feltéve, hogy az formátumozás nélküli tiszta ASCII fájlt generál) konfigurálhatjuk a PAK programot arra is, hogy az archív megjegyzést mindig az általunk megadott állományból fűzze hozzá az archívhoz.

— Saját parancs- és kapcsolókészletet állíthatunk össze gyakoribb feladatainkhoz.

Hogyan lássunk neki?

1. Hozzunk létre egy (lista)fájlt, ami a (ZIP)kommentet tartalmazza majd, mondjuk a C:\ZIP könyvtárban FILEFEJ.DAT néven. Ebben készítsük el azt a logót, amelyiket a (ZIP)kommentbe szánunk. A fájl első sora legyen üres vagy legfeljebb 55 karakter hosszú (a PKZIP és a PKUNZIP ugyanis a kommentet nem a sor elejétől kezdi írni).

2. Ha még nem lenne a pathban, tegyük oda a PAK.EXE programot. Ha szükséges, konfiguráljuk a PAKINST programmal.

3. PAK h /z ZIPfájl.ZIP

(H a ZIPkomment bekérése, /z a ZIP forma megőrzése. Meg kell adni a kiterjesztést is, mert különben az azonos nevű ARC és PAK állományok is módosulnak.)

4. A program bekéri a ZIPkomment szövegét. Ide egy bevezető @ karakter után be kell írnunk a ZIPkommentet tartalmazó állomány teljes nevét (a fenti példánál maradva: @C:\ZIP\FILEFEJ.DAT).

5. A fájlkommenthez a PAK r parancsa vagy az /r kapcsoló való:

PAK r ZIPfile.ZIP

Az eredeti PAK ilyenkor a ZIPkommentet is módosítja, de ez javítható (lásd alább). Használható még a PAK c /z /r ZIPfájl.ZIP fájlnév.kit szerkezet is a fájlkommentek bevitelére.

Haladóbbak számára ajánlom, hogy a PAK programhoz definiáljanak új parancsokat és kapcsolókat saját feladataikra. Ennek a mikéntjéről bő információ található az októberi Compfairre az Alaplap Könyvek sorozatban megjelenő „Tömör gyönyör, avagy a DOS alatt működő PC-k adat- és program-tömörítési lehetőségei" című könyvemben.

A bővített ZIPkomment

— Nem teljesen szabványos voltuk nagy ritkán megzavarhatja az archív kezelő segédprogramok egy részét, így a ZIPVIEW programot is. A kibővített ZIPkommenttel ellátott állományoknak már a tartalomjegyzékét sem írja ki.

— Tapasztalatom szerint a ZIPDMP nevű szeletelőprogram sem kedveli túlságosan a bővített megjegyzéseket.

Ilyen esetekben esetleg segíthet a felesleges ZIPkomment eltávolítása a PKZIP –z parancsával (a ZIPkommentet bekérő üzenetre legalább egy szóközt be kell írnunk az <ENTER> megnyomása előtt, amivel felülírja a PKZIP az előző kommentet). Ha még mindig gondot okoz az adott ZIPfájl használata, akkor az azonnal kéznél levő PAK program CONVERT parancsával javíthatunk esetleg rajta, ha ez sem segít, akkor újra kell csomagolnunk a kérdéses ZIP állományt.

SNGLCMNT.EXE

Egy egészen használható programot találtam, ami egy átlagos kommentezési feladatra majdnem olyan jó, mint a PAK. Igaz, hogy a SNGLCMNT.EXE nem szövegszerkesztővel készített állományból veszi a kommentet, viszont a program szerkesztő ablakában (12 sorban maximum 75-75 karakter) tetszésünk szerint készíthetünk kommenteket a ZIP fájlhoz. A ZIP archívot megadhatjuk már a parancssorban, vagy paraméter nélkül is indíthatjuk a programot.

Nagy Gábor

INFORMÁCIÓKÉRÉS: 28 ▼

NETREND RT

1089 Budapest, Elnök u. 1.
Tel: 113-8217; 133-4760 • Fax: 113-9537

XT-10 számítógép	
– 640 kilobájt RAM	
– Multi I/O kártya	
– 360 kbájtos FDD	
– 101 gombos billentyűzet	29 800
– 84 gombos billentyűzet	3 500
– 101 gombos billentyűzet	3 800
AT 286-12/16 számítógép	
– 1 megabájt RAM	
– FDD/HDD vezérlő + S/P kimenet	
– 1,2 Mbájtos FDD	
– 101 gombos billentyűzet	37 950
AT 286-16/21 számítógép	42 950
NEAT 286-16/21 számítógép	46 950
NEAT 286-20/26 számítógép	54 550
NEAT 286-24/32 számítógép	58 650
Hálózati terminál:	
NEAT 286-12	**48 750**
NEAT 286-16	**56 500**
AT-386-20/25 MHz,	
– 2 megabájt RAM	
– FDD/HDD vezérlő + S/P kimenet	
– 1,2 Mbájtos FDD	
– 101 gombos billentyűzet	85 800
AT 386-25/33	**91 700**
AT 386-25/43, 64 kB cache	**103 400**
AT 386-33/58, 64 kB cache	**110 000**
486-25/117, 4 MB RAM	
– 128 kB cache	248 000
486-33/147, 4 MB RAM	
– 128 kB cache	290 050
Monitor-csatolókártyák:	
Monokróm	1 700
CGA	2 400
EGA	5 400
VGA 800x600	9 600
VGA 1024x768	11 800
VGA 1024x768	15 600
VGA 1024x768	16 500
Monitorok:	
Egyszínű (borostyánsárga)	10 900
Egyszínű (papírfehér)	11 500
EGA	29 500
VGA (1024x768)	35 500
VGA egyszínű, 1024x768	22 000
VGA Multisync	44 500
MFM, ESDI és IDE winchesterek nagy választékban!	
Szünetmentes áramforrások:	
UPS 400 VA NOVELL	39 900
UPS 550 VA	32 000
UPS 600 VA NOVELL	45 800
UPS 1 kVA	54 600
UPS 1,2 kVA NOVELL	98 500
UPS mon. kártya	7 500
Nyomtatók:	
FX-850	49 500
FX-1050	48 750
LQ-850	76 400
LQ-2500+	129 000
DFX-500	183 500
DL 5600	195 000
HP LASERJET III.	199 000
Memóriabővítő kártyák:	
286/2 megabájt	9 900
286/3,5 megabájt	10 800
386/2/8 megabájt	15 000
RAM-ok:	
4164-10	140
41464-08	290
41256-08	160
41256-06	280
44256-08	780
511000-10	750
511000-08	760
SIM/SIP 1 MB	5 690
Koprocesszorok:	
80287-10	14 000
80287-20	44 000
80387-20	44 000
80387-25	52 800
80387-33	69 900
Modemek:	
2400 baud belső	10 900
2400 baud külső	14 500
2400 baud MNP-5	18 500
Telefax/modem:	
9600/2400 baud	32 500
8 felhasználós hálózati telefax	75 000
Egerek, scannerek:	
GM-6000 egér	4 950
Microsoft egér	16 000
Logitech soros	5 440
Logitech scanner	21 700
Catchword karakter-felismerő program	22 900
HP SCANJET PLUS	259 000
Plotterek:	
SEKONIC 450	115 600
HP 7475A A3	179 000
MUTOH 910E	1 380 000
ARCnet kártyák:	
8 bit LIN DATA	4 800
8 bit ZOT	5 400
8 bit SMC	9 900
16 bit LIN DATA	9 900
16 bit ZOT	10 500
ETHERNET kártyák és tartozékok:	
8 bit NE-1000	9 900
8 bit DE-100	14 500
8 bit DE-150	19 600
16 bit NE-2000	12 900
16 bit DE-200	16 500
ARCnet-kiegészítők:	
Passzív HUB	2 000
Aktív HUB (int 4)	7 800
Aktív HUB (ext)	14 000
Aktív HUB + kártya	12 000
Csatlakozókábel	1 500
BOOT-EPROM	2 000
Ethernet-kiegészítők:	
Transceiver	35 000
Transceiver BNC	25 000
Repeater (2 port)	92 000
Repeater (4 port)	148 000
Hálózati szoftverek:	
Novell NetWare	
V.2.2/5	72 000
V.2.2/10	161 000
V.2.2/100	444 000
V.3.11/20	282 000
V.3.11/100	565 000
V.3.11/250	1 010 000
NACS	110 000
Asyncron Remote Bridge Program	29 600
D-Link Lansmart op. rendszer	34 500
D-Link Bridge	29 600
ACS D-Linkhez	28 000
Remote Access	19 600
Screen monitor	15 000
LAPTOP-ok:	
LT-3400 (NEAT) 40 MB HDD	199 000
CP-8100V (386) 100 MBB HDD	299 000

És még sok minden egyéb...
Keresse termékeinket Székesfehérváron, az IZISZ Kft.-nél is!
Székesfehérvár, Palotai út 139. Telefon: (22)16-049.
A Netrend Rt. a Novell Inc. hivatalos dealere.
Vállalkozunk komplett hálózati rendszerek szállítására, igény szerinti kiépítésben.
Komplex rendszerfelügyelet (hálózati is), szaktanácsadás, hardver- és szoftverkarbantartás.
CAD, DTP rendszerek kiépítése, szükség esetén üzemeltetése. Kérje részletes tájékoztatónkat!
Áraink az ÁFA-t nem, de a 6 hónap csereszavatosságot tartalmazzák, egy év csereszavatosság — plusz öt százalék. Készpénzfizetés esetén öt százalék kedvezmény!
Önkormányzatok, oktatási intézmények, egészségügyi szervezetek részére 5 százalék kedvezmény!
Kedvező lízingfeltételek!

INFORMÁCIÓKÉRÉS: 14 ▲

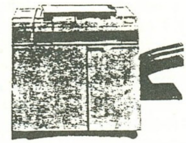
INFORMÁCIÓKÉRÉS: 06 ▼

Canon

VÁLTSON SZÍNESRE Canon MÁSOLÓGÉPPEL!

CSÚCS AMIT TUD:

- Kicsinyítés
- Nagyítás
- Montírozás
- Tükörkép-készítés
- Képismétlés
- Poszter készítés
- 17 millió színáranyalat



Győződjön meg róla!

fénymásolás

TONER KFT

1095 Budapest, Mester utca 21.
Tel.: 113–1687, 134–3516

INFORMÁCIÓKÉRÉS: 19 ▼

Ingres

VILÁGSZÍNVONAL HAZAI ÁRAKON

Ideális fejlesztőkörnyezet és futtatórendszer adatbázis-alkalmazásokhoz:

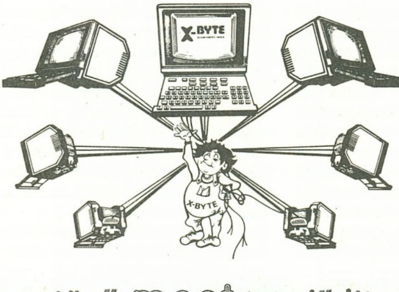
- Komplett SQL adatbázis-kezelés.
- Negyedik generációs fejlesztőkörnyezet.
- Interaktív maszkeditor.
- Menü- és ablakkezelés.
- Tökéletes adatbiztonság.
- Tranzakció-kezelés.
- Nagy hatékonyság.
- Server–kliens architektúra.

Újdonság!

Grafikus alkalmazásgenerátor (Windows 4GL) az SCO Open Desktop rendszeréhez.

Felvilágosítás:
VT-SOFT KFT.
Telefon: 180-3744
Telefax: 180-3750

INFORMÁCIÓKÉRÉS: 17 ▼



a jövő most kezdődik!

X-BYTE®

SZÁMÍTÁSTECHNIKA
Törv. védve

SZÁMÍTÓGÉP HÁLÓZATOK
Kiváló minőség, közepes ár!

kiváló áruk fóruma

1138 Budapest, Népfürdő u. 17/E
Tel. és fax: 173-1232
Telex: 22-3399

Toprekiám

INFORMÁCIÓKÉRÉS: 23 ▼

A

BAREX COMPUTER

SZAKÜZLET KÍNÁLATA:

ST-157A winchester	16 200.–tól
210 M Quantum	70 000.–
Handy Scanner 400 DPI, 105 mm	12 000.–
80287-10 processzor	11 000.–
GM-6 mouse	1 400.–

3M floppylemezek:

DS DD 5,25"	752.–
DS HD 5,25"	960.–
DS DD 3,5"	1 300.–
DS HD 3,5"	1 950.–

Áraink 12 hónap garanciával,
ÁFA nélkül értendők.

1054 Budapest V., Bajcsy-Zsilinszky út 54.
Telefon: 131-0946, 111-6025

Átadható információk
Step by step

A STEP
— Standard for the Exchange of Product Model Data — rendeltetése szerint a jövőnek készülő általános célú adatátviteli szabványegyüttes, amely termékmodelleket leíró információk különböző CAXX rendszerek között átadható szemléltetési formájának meghatározására irányul. Olyan adatformátumot és -struktúrát specifikál, amely az adatok azonos idejű és archivált feldolgozásában egyaránt előnyösen használható.

A STEP projektet az amerikai IGES/PDES Organization PDES-sel karöltve az ISO TC 184/SC4/WG1 munkacsoportja koordinálja. Az első munkaváltozat 1988 végére készült el. Már ezen is látszott, hogy a STEP nem egy-ember léptékű alkotás, mivelhogy a részben kész dokumentációja is több ezer oldalt tett ki — ami nagyobb részben formális definíció, kisebb részben alkalmazási protokoll.

Módszer minden mennyiségben

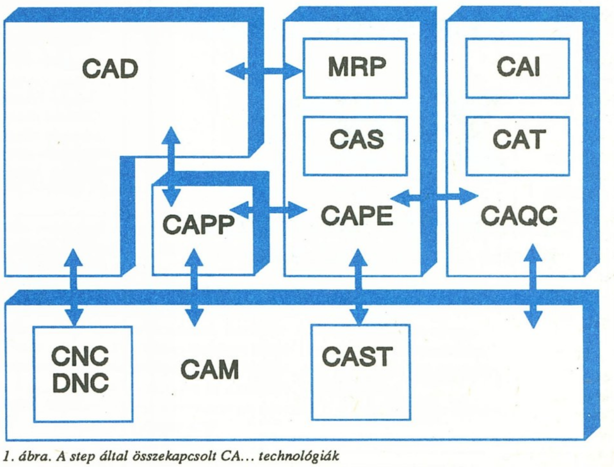
A szabványt több rész alkotja, ezek mindegyike egy-egy önálló témakört fed le. E cikk megírásakor még nem állt rendelkezésemre a kész ipari szabványjavaslat, így ennek csak tervezett fejezeteit áll módomban ismertetni: 1. rész: Áttekintés, 11. rész: Az EXPRESS nyelv, 21. rész: A fizikai fájl, 31. rész: Megfelelőségi vizsgálat, 41. rész: GPDM — Általános termékadat modell, 42. rész: Alakszemléltetés, 46. rész: Ábrázolás és megjelenítés, 101. rész: Rajzi erőforrások, 201. rész: Rajzi alkalmazási protokoll.

Minden rész önmagában teljes, önálló szabványt képvisel. Ez lényeges eltérés a korábban megismert IGES-hez képest, ami monolit egész (pontosabban a hatodik változata kapcsán merült fel a definíciók és az alkalmazások különválasztása). További, jelenleg kidolgozás vagy ellenőrzés alatt álló, de 1992-ben megjelenésre szánt részek a következők: 12. rész: Szervezési keret, 44. rész: Termékstrukúra, 45. rész: Anyagok, 48. rész: Alaksajátosságok, 105. rész: Kinematika, 2XX. részek: További alkalmazási protokollok.

Teljes kiépítettség esetén a CAXX rendszerek kapcsolt hálózata a termék létrehozásának teljes folyamatát lefedi (1. ábra). Hogy a különböző rendszerek között az adatok korlátozás és szakadás nélkül mozgathatók legyenek, átfogó adatkommunikációt kell szabványosítani. Ehhez a termékmegvalósítási környezet egészét modellezni kell, amit a STEP három lépcsőben valósít meg. A legáltalánosabb információs struktúrát meghatározó modell az integrált termékinformációs modell, amit az angol megnevezése alapján egyszerűen IPIM modellnek neveznek. Az IPIM az alkalmazások, az erőforrások és az életciklus részmodellekre bontható. A jelenleg legnagyobb mélységben kidolgozott erőforrás részmodell a geometriai, a topológiai, az anyagokra vonatkozó, az alakleírási, az alaksajátosság, az ábrázolási és a tűrésezési információkat öleli fel.

Míg az erőforrás és az alkalmazás részmodellek segítségével csak a késztermék mindenkori állapotát tudjuk leírni, addig az életciklus modell a termékfejlődést kronológiailag is kezelni tudja. Megadhatók például a gyártási terv, a kiindulási (nyers) állapot adatai, az NC-megmunkálás közbeni és ellenőrizendő végtermékre vonatkozó adatok. E lehetősége a STEP szabványt az általános minőségbiztosítási rendszer eszközévé is teszi. A valamely konkrét termékhez tartozó általános információkat az átfogó termékdefiníciós modell (GPDF) foglalja magában, a termékekhez kapcsolódó alkalmazásspecifikus információkat pedig az alkalmazási adatokat leíró protokollok (ADPM) határozzák meg. A STEP hatáskörében jelenleg a gépészeti, az elektronikai, az elektrotechnikai és az építőmérnöki alkalmazásokat fedi le.

A V1.0 változatban megjelenő alkalmazási protokollok várhatóan a 3D



1. ábra. A step által összekapcsolt CA... technológiák

rajzolást, a termékadatok felépítéstől függő kezelését, a testek B-rep szemléltetését és a szoborfelületek leírását foglalják magukban. Hozzá kell tenni, hogy az alkalmazási protokollok lényegében a STEP funkcionalitását terjesztik ki. Sajátosságuk, hogy lefedik például a tervezés, az elemzés, a gyártás, az ellenőrzés, a karbantartás... életciklusok alkalmazástól függő információigényét.

Rendeltetését tekintve a termékmodell olyan információállomány, amely a termék létezésének valamennyi állapotát leírja. A leírás egyrészt figyelembe veszi a jelenlegi rendszerek lehetőségeit, másrészt tekintettel van a jövőbeli rendszerek igényeire. A STEP kidolgozóinak célkitűzése az automatikus, emberi közreműködést egyáltalán nem igénylő adatforgalom megvalósítása volt. Minden termékmodell zárt információs csomagot alkot. Ismét eltérően az IGES specifikációtól a STEP három rétegre oszlik: az alkalmazási, a logikai és a fizikai rétegre (2. ábra). Az alkalmazási réteg egy adott alkalmazási területre vonatkozó protokollokat írja le formális eszközökkel. A logikai réteg a fogalmi struktúrákat határozza meg és egységesíti. A fizikai réteg a termékmodell-adatfájlok létrehozásának szabályait adja meg.

A STEP egységes adatleíró nyelvet alkalmaz, az EXPRESS-t, ez az alapja a formális definiálásnak. Az EXPRESS — és egyben a fizikai fájl — nyelvi formátuma a WSN (Wirth Syntax Notation) jelölésrendszeren alapul. Az EXPRESS nyelv őse egy viszonylag egyszerű adatstruktúrákkal dolgozó grafikus IDEF1X leíró nyelv volt. Legfőbb jellegzetessége, hogy a Pascal nyelvből közismert típusdeklarációkat alkalmaz. A típus valamely statikus — azaz időben nem változó — struktúra kijelölésére szolgál. Az EXPRESS emellett altípus-definiálást is lehetővé tesz, ami öröklődési mechanizmusok megvalósítását támogatja (4. ábra). Az adatszerkezetekben értékek és értékhivatkozások egyaránt megadhatók, sőt az elfogadható értékek tartományának megszorítására is lehetőség van. Az alapegység-definíciókhoz értéköröklődés is hozzárendelhető.

Ami átjön és ami átmegy

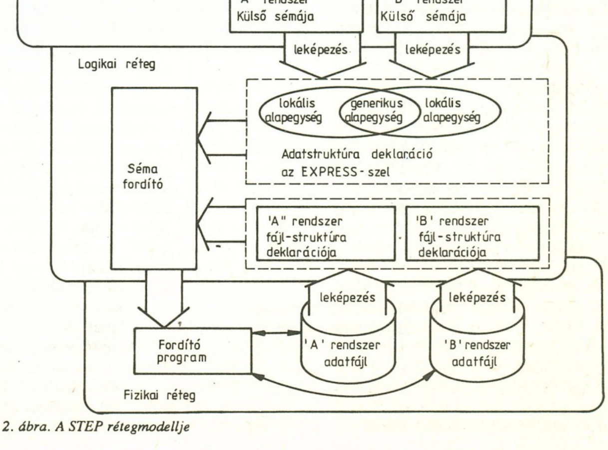
Az EXPRESS logikai adatszerkezeteinek a STEP-beli fizikai fájlok adatállományára való leképezése közvetlenül történik, ugyanis az alaptípusoknak létezik a fizikai fájlbeli megfelelője, a lista-, tömb- és halmazstruktúrák pedig a fizikai fájlban értelmezett listatípusra képezhetők le. A STEP-beli adatkommunikációs fájl nem bináris fájl, hanem karakteregyüttesekből álló tokenek sorozata. Moduláris felépítésű, az információt közvetítő logikai elemek az alapegységek. A szintaktikai egységek elválasztására határolókat kell alkalmazni. Az EXPRESS nyelv meghatározza azokat az adatstruktúra-elemeket, amelyek megjelenhetnek a STEP-fájlban. A fizikai fájl lényegében ezeknek az elemeknek és részstruktúráknak az összességéből képződik. A fájl tartalma struktúrameghatározó szimbólumokból, nyelvi definíciós alapegységekből, termékleíró adatokból és nyomtatásvezérlő elemekből áll. Ez utóbbi elemek lehetővé teszik a nyomtatást, illetve formátumának meghatározását. A STEP-fájl szekvenciális szervezésű, viszont nincs kikötés arra, hogy a fájl egy menetben feldolgozható legyen. Az egyes rekordok szabad formátumúak, tehát a rekordon belüli helyfoglalásnak nincs jelentősége. Ebből adódik az is, hogy a rekordok változó hosszúságúak is lehetnek.

A fájl tartalma papír adathordozón kétféleképpen jeleníthető meg: közvetlen vezérlés alkalmazásakor a nyomtatási kép a fájlba beírt jelek hatására nem előrendezett fájltartalom esetén is kialakul. Ilyen vezérlőjel például az új sor /N/ vagy a laptovábbítás /F/. Közvetett vezérlésnél az adatok fájlba írásakor eleve betartjuk a formátum kialakítására vonatkozó szabályokat. Ilyen előírás például, hogy a sorok balra igazodnak, a fenntartott szavak, a TÖRZSRÉSZ szakasz alapegységei és az azonosítóként használt alapegységek mindig új sorba kerülnek és így tovább. Ezeket a STEP fizikai fájlra vonatkozó része tárgyalja.

Mint említettük, a STEP-fájl strukturált, jelenleg két szakasza rögzített, ezek a TÖRZSRÉSZ (HEADER) és az ADATRÉSZ (DATA). A szintaktikai formátumot a 3. ábra mutatja. A TÖRZSRÉSZ szakaszból pontosan egy lehet a fájl elején. Tartalmazza a fájl nevét, a létrehozás időpontját, a szerző nevét és hovatartozását, a fájl létrehozásakor követett STEP-specifikáció változatszámát, az alkalmazott előfeldolgozó és tervezőrendszer azonosítóját, a fájlbeli információk kinyeréséhez szükséges utófeldolgozó változatszámát, továbbá a felhasználó által definiált alapegységek példányait is. Ez utóbbi opcionális, és szintaktikai korlátozások érvényesek.

Mindkét szakasz bizonyos alapegységeket tartalmazhat, amelyek általános formátuma a következő: KULCSSZÓ (paraméter_lista). A TÖRZSRÉSZ szakaszban a fájlazonosító alapegység — (FILE_IDENTIFICATION('Példa','Dátum', ...); — használata szükséges. A szakaszok egyrészt standard alapegységeket tartalmazhatnak, másrészt beírhatók felhasználó által definiált alapegységek is. Ezek neve előtt a ! jelölés alkalmazandó. Az alapegység-példányok azonosítói a @ jelet követő számjegyek. Az alapegység-példányokra való hivatkozás a # jelet követő egész számokkal történhet, ha egyébként az azonosító szerepel a fájlban.

Az ADATRÉSZ szakasz az alapegységek beírását tekintve egységes sémá-



2. ábra. A STEP rétegmodellje

jú. Minden alapegységre annak azonosítóját, a típus megnevezését, valamint az attribútumokat kell megadni, egymástól vesszővel elválasztva. Az ADATRÉSZ sajátossága a szakaszon és alapegységen belüli érvényesség, azaz a hatáskör, amelynek struktúraképző hatása van, és a valamitől függő létezés kifejezésének eszköze. Például:

```
Alapegység típus kulcsszó
&SCOPE
.
. Alapegység-előfordulások
.
ENDSCOPE;
```

Az érvényesség funkció testében definiált alapegységek nem hivatkozhatók a hatáskörön kívüli alapegységekből, kivéve, ha az EXPORT_LIST alapegységgel elérhetővé tesszük. A STEP külső hivatkozásokat is lehetővé tesz. Az EXPORT alapegység más létező fájl megnevezett alapegységére mutat, a LIBRARY az alapegység helyét könyvtárként jelöli ki, míg teljes fájl hivatkozását az EXTERNAL_REFERENCE teszi lehetővé.

A STEP egy az adatkommunikációs előírásokkal eddig le nem fedett területre is kiterjed, mégpedig az alkalmazási protokollok kidolgozására. Látni kell ehhez, hogy a STEP gyakorlati alkalmazásának sikerességét az alkalmazási referenciamodellek nagymértékben elősegítik. A referenciamodell megadja az adott alkalmazásban használható leírósémákat, alapegységeket, adattípusokat, konvenciókat, kapcsolatkijelöléseket és így tovább. A STEP a gyakorlat szempontjából lehetőséget ad bármilyen számítógépes belsőábrázolású modell leképezésére. Az EXPRESS-t úgy tervezték, hogy a STEP-alkalmazás későbbi szakaszában is a jelenlegi információs modellek alapján elő- és utófeldolgozó szoftverek (processzorok) fejleszthetők legyenek. Eltérés az IGES-hez képest, hogy az egyes CAXX rendszerekhez a processzorokat nem szükségszerűen az adott rendszer fejlesztői dolgozzák ki, hanem tesztelést is végrehajtó programozási központok. Eddig mintegy másfél tucat — széles körben elterjedt — rendszerhez alakítottak ki processzorokat.

Az elhatározások tükrében a STEP-re alapozott információ-kommunikációnak négy egymásra épülő szintjét tervezik megvalósítani.

A fájlcseréét, a projekt-adatbázisok, a globális adatbázisok és a tudásbázis cseréjéét. Mindegyik szintre az IPIM alapján származtatják az információs struktúrákat.

Az adatformátum az egyes szinteken különböző lehet, leginkább abból eredően, hogy az egyes kiterjedtségi szinteken különböző célok és feltételek érvényesek.

```
ENTITY direction
        SUPERTYPE OF              (two_space_direction OR
                                  three_space_direction)
        SUBTYPE OF (vector):
END_ENTITY:
ENTITY two_space_direction
        SUPERTYPE OF (direction):
        x                         : REAL :
        y                         : REAL :
WHERE
        (x*x+y*y)    0 :
        definition_spacece(two_space_direction)=2 :
END_ENTITY :
ENTITY  three_space_direction
        SUBTYPE OF (direction) :
        x                         : REAL :
        y                         : REAL :
        z                         : REAL :
WHERE
        (x*x+y*y)    0 :
        definition_space(three_space direction)=3 :
END_ENTITY :
```

4. ábra. Altípus definiálása a STEP-ben

```
STEP;
HEADER;
.
.
.            /*  TÖRZSRÉSZ — a fájl egészére
.                vonatkozó információkat
.                tartalmazza                      */
.
ENDSEC;
DATA;
.
.
.            /*  ADATRÉSZ — a terméket leíró
.                adatokat tartalmazza             */
.
ENDSEC;
ENDSTEP;
```

3. ábra. A STEP-beli fizikai fájl szakaszai

Hol az a 2000 dollár?

Végezetül még néhány kapcsolódó információ. Tekintve, hogy a célok megvalósítását olyan cégek szorgalmazzák, mint a Boeing, a DEC, a Lockheed, a Martin Marietta, a McDonnall Douglas, a Prime, a Northrop, a Rockwell — hogy a USA Air Force-t ne is említsük —, nem lehet a dolog mellett közönyt színlelve elmenni.

Mivel korábban abban a szerencsében volt részem, hogy bepillanthattam a kulisszák mögé, szeretném e helyen közreadni azt az információt is, hogy a NIST (Amerikai Szabvány- és Technológiai Nemzeti Intézet) szervezésében a társult vállalatok mindegyike 50 000—200 000 dollárt fizet be évente, és 3 éven keresztül 2-5 embert teljes munkaidőben biztosítanak a kutatási és fejlesztési munkában való közreműködéshez. Mivel hazánk elvileg résztvevő (P) státuszú taggá lépett elő az elmúlt évben a korábbi megfigyelő (O) pozíciójából, magyar képviselőnek is lehetősége van a részvételre — megközelítőleg 2000 dollárért (az utazást is beleszámítva).

Áttekintésem bizonyára hosszabbra sikeredettt volna, ha ez a „néhány" zöldhasú az én pénztárcámban lapul...

Horváth Imre

Nyílt titok

A PC szereptévesztése

A magyarországi számítógépparkra a COCOM-embargó furcsa, torzító hatást gyakorolt: korszerűtlen, régi IBM-gépeket utánzó, általában csak lassú, kötegelt feldolgozásra képes nagyszámítógépek és rendszerint silány minőségű személyi számítógépek egyvelegét eredményezte. Ahogy a középvállalat hiányzik a gazdaságból, úgy hiányoznak a többfeladatos, többfelhasználós (multiuseres, multitaskos) közepes és minigépek. A fejlett ipari országokban a vállalatok, bankok, kormányhivatalok stb. információs rendszerének gerincét ezek a gépek adják. Jelen gondolatok a kis-, közepes és nagyobb cégek informatikai igényei köré csoportosulnak, s mivel ezek kielégítése hálózatokban „cirkuláló" adatbázisrendszerekkel függ össze, végül is ebben a rovatban vannak talán a legautentikusabb helyen lapunkban.

Szegény ember vízzel főz, mondja a közmondás. A magyar gyakorlatban ez azt jelentette, hogy lassan minden feladatot a PC-kre testáltak: amit PC-vel (vagy jobb esetben ilyenek hálózatával) nem lehetett elintézni, az megoldhatatlan számítástechnikai feladat.

Őszintén szólva, tényleg bámulatos, hogy mi mindent csinálnak ma Magyarországon PC-vel. Az elfojtott vállalkozókedv 1981 óta (ekkortól lehetett gmk-kat alapítani) a számítástechnikában tört ki leghamarább. Ez természetes is, hiszen olyan új termékekkel foglalkoztak, amelyek az akkori állami iparban nem léteztek. Csekély tőke, viszont magas szakmai tudás volt szükséges, ezért a fiatal értelmiségnek ideális volt ez a pálya. Később persze a kereskedelem vált uralkodóvá ezen a területen, de most, hogy a hazai piac fizetőképes keresletének korlátaihoz elérkezett a számítástechnikai ipar, ismét előtérbe került a szakértelem.

Csúzlival elefántra

Minden káprázatos eredmény ellenére ma elmodhatjuk, hogy a PC-k szereptévesztése általános az országban: ahelyett, hogy tényleg személyi számításokra, levelezésre, a nagygépeken tárolt információk megjelenítésére használnák, a PC-k és hálózataik alkotják magát az információs rendszert. Erre azonban igazából sem teljesítményük, sem megbízhatóságuk alapján nem képesek.

A statisztikák árulkodnak a helyzetről: a lakosság létszámához viszonyítva 10-15-ször nagyobb a személyiszámítógép-sűrűség a fejlett ipari államokban, mint hazánkban; és legalább annyi miniszámítógép üzemel ott, mint ahány PC-hálózat nálunk.

Kérdezhetnénk — és mivel a PC-hálózatokon kívül más megoldást nem ismernek, sokan meg is kérdezik —, hogy miért baj az, ha PC-hálózatok képezik a vállalati információs rendszert? Érvként felhozzák, hogy a legkedvezőbb ár/teljesítmény mutatóval a PC-k rendelkeznek (a nagy sorozatok miatt), és fejlődésük töretlen: egyre gyorsabb és gyorsabb gépek jelennek meg.

Ha ez így igaz lenne, akkor nem sorjáznának a panaszok: lassú a gépem, elvesznek az adatok a hálózaton, összetörtek az indextáblák, rejtélyes leállások nehezítik az üzemeltetést, elvírusosodott a Novell hálózatom stb. Aki nem csak szép, új feladatok tervezésével tölti az idejét, az jól tudja, hogy tajvani PC-s hardverrel, MS-DOS és Novell szoftverrel, hazai rohammunkában készült, egyedi fejlesztésű alkalmazói szoftverekkel rendszert üzemeltetni nem egyszerű dolog.

Ezek az üzemeltetők naponta hallják a piacgazdaságra való áttérés közben meghajszolt menedzserektől: kevés és megbízhatatlan az általuk szolgáltatott információ. Tudjuk, hogy a szervezetek általános szervezettségi szintje alacsony. A számítástechnika azonban a szükséges precizitásával magával hozza a szervezettséget is: ezért különösen kívánatossá vált a fejlesztése. A minigépeket nélkülöző fejlődési irány azonban ez esetben zsákutca: nem lehet integrált vállalati információs rendszert, több százezer vagy több millió rekordból álló adatbázisokat PC-vel, PC hálózattal kezelni.

Az „üveggyöngy"-jelenség

A hónap témájában (18. oldal végén) olyan buktatókról tettünk említést, amelyeket el kellene kerülni a magyar számítástechnikai fejlődés mostani kritikus szakaszában. Bővebb magyarázatra az ott utolsóként említett veszélyforrás szorul. Lényege az, hogy a szép, színes grafikus rendszerek sokkal jobban eladják magukat, mint a hatalmas háttérmunkával megszervezhető információs rendszerek, adatbázisok. Ám vegyük észre, hogy a fejlett ipari államokban ezeket a grafikus rendszereket már működő adatbázisok megjelenítésére használják. Nálunk sok programfejlesztő a szoftver elkészítését a bejelentkező képernyő megírásával kezdi; és miután a rendelkezésre álló idő fele ezzel elment, az alapadatokat biztosító adatkezelő programrészt összecsapja. Bármilyen ingatag is az alap, a szép külső — sajnos még mindig — eladja.

Ezen a ponton érdemes kitérnünk arra, hogy a csapda mindhárom torkának megnyílása, vagyis a veszélyforrások keletkezése egyetlen végső okra vezethető vissza. Ez pedig a szükségesnél alacsonyabb általános képzettségi szint a számítástechnikai szakemberek között. Különösen gyenge a logisztika: lámpással kell keresni a relációs adat-

bázisokhoz értő szervezőket. Kevés az olyan programozó is, aki (teammunkában) több százezer vagy még több soros szoftverek megírásához értene. Nem jobb a helyzet az üzemeltetők között sem: nem ismerik az operációs rendszereket, és nincsenek tisztában az adatvédelmi intézkedések fontosságával.

Gyors javulás a képzettség területén nem várható, mert a szakmában dolgozók létszáma folyamatosan nő, és az intézményi képzés ezzel ma nem tart lépést. A munkanélküliek közül csak kevésnek van olyan felkészültsége, amely ezen szakmákra történő átképezhetőség feltétele.

Az ütőképes stratégia

Mindezek után milyen tanácsokat lehet tehát adni ma egy adatokra éhes, kicsinél nagyobb vállalat, vállalkozás menedzserének, hogyan is fogjon hozzá a számítógépesítéshez? Az alábbi egyszerű receptet ajánljuk:

— Tegye rendbe a szervezetet: a munkakörök, a felelősségi viszonyok, az adatszolgáltatási kötelezettségek legyenek világosak.

— Vegyen fel vagy bízzon meg egy jónevű vezető szervezőt, aki megálmodja a rendszerhez az adatbázist. A dokumentációt előre kérje, és magyaráztassa el a folyamatokat, amelyekben megtestesül a rendszer működése. Ez a legkritikusabb pont: az az adat, amelyik nem szerepel az adatbázisban, soha nem lesz megtalálható az információs rendszerben.

— Válasszon az adatbázis méretének megfelelő nagyságú gépet UNIX-szal, mert az ún. nyílt rendszerek koncepciójának ez a megfelelője az operációs rendszer szintjén.

— A vezető szervező irányításával készíttesse el a programok specifikációját, majd vizsgálja felül azokat, hogy tényleg megfelelnek-e a szükségleteknek.

— Írassa meg a programokat (lehetőleg valamilyen negyedik generációs programozási nyelven). Itt már csak azt kell ellenőrizni, hogy a programok megfelelnek-e a specifikációknak.

— Ne feledkezzen meg a fejlesztőktől megrendelni a bevezetést, az oktatást, a kezdeti rendszerfelügyeletet és a későbbi folyamatos tanácsadást.

— Különös gonddal válassza ki a rendszergazdát, és ne sajnálja a rendszergazda képzésére fordított összegeket.

A terep nehéz, de gazdag hozamú

Szándékosan nem jelöltünk meg a receptben semmiféle kész szoftvert — kivéve a UNIX-ot mint alapot. Sajnos jelenleg fehér holló ritkaságú az olyan alkalmazási szoftver, amely nagy adatállományokkal is működőképes, és hatékonyan paraméterezhető. Nyugaton vannak ilyenek, de hiányoznak a magyar nyelvi interfészek; és az ilyen szoftvereket kínáló társaságok közül ma még egyik sem jeleskedett a magyarításban. Ez természetesen nem jelenti azt, hogy egyes jól használható modulokat ne készen vegyünk meg, de ennek a rendszerbe illesztéséről továbbra is egyedileg kell gondoskodni. És főleg: ne dőljünk be annak, amikor a konkurencia a PC-ire kifejlesztett (és éppen kidobás előtt álló) célszoftverét kínálja nekünk potom pénzért — mert egy számítástechnikai rendszer bevezetésekor a legnagyobb összeg mindig a bevezetés; a dolgozók oktatása, a munkaidő-kiesés és a túlórák, az első adatbázis-létrehozás.

Az ajánlás ugyancsak nem említ semmiféle grafikus interfészt. Első megközelítésben ez nem szükséges, először a biztos alapokat kell lerakni. A vállalatoknál, intézményeknél a „látványos" PC-bőség ellenére ezek a legtöbb helyen még hiányoznak. Egy olcsó, 35-40 ezer forintos monokróm terminál a legtöbb helyen megfelelő. (Figyelem! Ergonómiailag gondosan tervezett, egész napos munkára szolgáló eszközökről van szó, nem pedig vibráló monitorú, a legalacsonyabbnál is alacsonyabb árfekvésű PC-kről!) Természetesen a meglévő PC-kről, illetve ezek hálózatáról sem kell lemondani: egyéni nyilvántartásokra, levelezésre, egyedi kalkulációkra és a nagy háttéradatbázis részeinek megjelenítésére ezek továbbra is jól használhatók. A nyílt rendszereket eleve úgy tervezték, hogy a PC-k, PC-hálózatok azokhoz csatlakozni tudjanak.

Ha a felsorolt tanácsoknak megfelelően járnak el a menedzserek, akkor 1-3 év alatt megfelelő szintű számítástechnikát tudnak megalapozni cégüknél. Ennél gyorsabban ritkán érhető el eredmény. Nyugat-Európában is 2-5 évet vesz igénybe a kész szoftverek bevezetése. Az ottani, valamivel rutinosabb környezetekben is az a tapasztalat, hogy a dolgozók fogadókészsége általában nem teremthető meg ennél rövidebb idő alatt.

Polló László

INFORMÁCIÓKÉRÉS: 21 ▲

INFORMÁCIÓKÉRÉS: 03 ▲

Ez a helyes írásvetítő!

Nappali fény mellett is kitűnő képet varázsolnak a vászonra a Polaroid írásvetítők.

A könnyen hordozható legkisebb modell mindössze 4,5 kg.

Előadások látványossá tételéhez ideális útitárs!



Többféle változatban kaphatók a

FLOPPYLAND

számítástechnikai szaküzletben
(Budapest V., Váci utca 84. Telefon/Fax: 118-2651)
és országszerte a Cédrus Rt. viszonteladóinál.

Polaroid

INFORMÁCIÓKÉRÉS: 29 ▲

Bill Gates a rendszerek jövőjéről



Bill Gates Amerika legfiatalabb „self-made" milliárdosa. Ő a Microsoft sikerek mozgatórugója. Legújabb büszkesége a Windows 3. (Idén májusban jelent meg, és a PC történelem leggyorsabban fogyó szoftvere lett.) Alig 35 évesen lábai előtt hever a világ, mindent elért az eddigiekben, amit akart. A világ legsikeresebb szoftvercégének egyik alapítója, vezetője már 15 éve. Tekintélyére jellemző, hogy az IBM, Apple, AT&T, Santa Cruz Operation cégeknél minden mondatát jegyzik, amely a DOS, OS/2, Windows, a Macintosh szoftverpiac vagy a Unix rendszerek témakörében elhangzott. Tulajdonképpen mindenki odafigyel, amikor ő nyilatkozik.

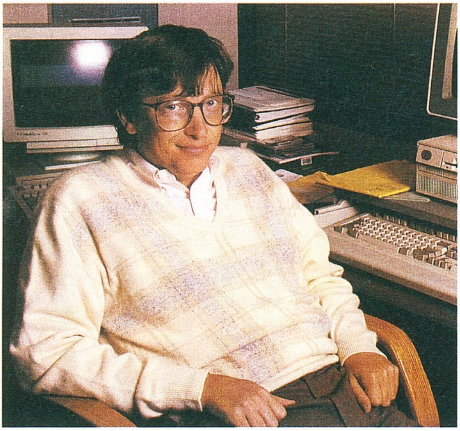
Ez az egyedülálló karriertörténet csaknem húsz éve kezdődött. Az akkori középiskolás Gates, barátjával, Paul Allennel programokat írogatott, mintegy zsebpénz-kiegészítésként. Ebben még nincs semmi rendkívüli. 1974-ben, amikor már a Harvard Egyetem hallgatója volt, Allennel úgy gondolták, hogy nagy valószínűséggel a Basic lesz a jövő nyelve. Ki is fejlesztettek egy Basic-et az akkori első mikrogépre, az MITS Altair-re. A fejlesztés sikerén felbuzdulva megalapították a Microsoft céget. Az igazi sikertörténet azonban hat évvel később kezdődött. Jó kapusnak szerencséje van — mondják. Velük is valahogy így történt. Az IBM először ugyanis nem az ő DOS operációs rendszerüket akarta felhasználni PC-jeihez, hanem a Digital Research cég CP/M rendszerét. Annak köszönhetően, hogy a Digital Research és az IBM nem tudott megegyezésre jutni, a Microsoft megindult meredeken felfelé ívelő pályáján a sikerek csúcsa felé. Ők alkotják a szabványt a programozási nyelvek, az operációs rendszerek, az alkalmazói szoftverek területén. Alapító társa, Paul allen azóta kivált a Microsoft-ból, saját céget hozott létre Asymetrix néven. Egyik fejlesztésük a Windows 3-nak részét képezi. Fantasztikus évet tudhat maga mögött a Microsoft: 1990-re 1,18 milliárd dolláros bevételt könyvelhettek el, s ez 47%-os növekedés az 1989-es 803 millió dollárhoz képest.

Érthető módon a számítástechnikában mindenki megkülönböztetett figyelemmel kísérte Bill Gates szavait Las Vegasban, az 1990. évi Comdex kiállításon. Előadásában elmondotta, hogy szerinte a jövőt egyértelműen az egyre inkább emberközeli gépek jelentik. Ennek megfelelő vezérfonal jelenti a Microsoft stratégiáját is. A 80-as években olyan számítástechnikai szabványt alkottak, amelynek következtében az alacsony árkategóriájú gépek elárasztották a világot. Szinte mindenki hozzájuthatott egy-egy géphez. Mára már a PC tulajdonosok száma megközelíti a 60 milliót. Jelenleg a PC szoftverek felhasználói olyan új korszak elé néznek, amelyben a desktop alkalmazások és a valóban hordozható PC-k természetessé válnak. Céljuk olyan szoftverek kifejlesztése, amelyek miatt senkinek sem kell eldobnia eddigi fejlesztéseit, az új kor követelményeinek mégis megfelelnek. Objektumorientált állományrendszerekkel dolgoznak, kezelik az osztott álományrendszereket, felismerik a kézírást és több különféle médiát kevernek. Jelenthet ugyan rövid távon előnyöket egy-egy nem szabványos alkalmazás, de feltétlenül az a jobb megoldás, ha az új technológiákat a PC architektúrájába, illetve a rendszerszoftverbe integrálják. Célszerű tehát egységesíteni az életünket kísérő információkat. Ha a szabványokat sikerül nagyon körültekintően kell kidolgozni, akkor minden felhasználó önszántából követni fogja azokat. Idén év elején a Personal Computer Word kérdéseire válaszolva számos meghatározó jellegű jövendölés hangzott el Bill Gates szájából. Először az operációs rendszerek, nyílt rendszerek témakörével kapcsolatban fejlette ki véleményét.

Érdekes helyzetet teremtett a Windows fergetes sikere. Eddig ugyanis a DOS és az OS/2 versenyében állandóan arra hivatkoztak a DOS pártiak, hogy nincs megfelelő szoftvertámogatás az OS/2-höz, az eddigi DOS környezetű alkalmazásokat pedig senki nem akarja kidobni azért, mert OS/2-re tér át. Itt lép be a Windows. Eltolódik a hangsúly a Windows felé, a DOS rovására. Windows környezetben grafikus alkalmazások születnek. OS/2-höz készült Windows fejlesztéseinket könnyűszerrel árvihetjük és futtathatjuk OS/2 Presentation Manager alatt. Ekkor pedig minden OS/2 nyújtotta előnyt kihasználhatunk. Megkönnyítik tehát a Windows fejlesztői az első lépéseket a DOS-ból felfelé. A DOS és a Windows együttesét már egy középméretű operációs rendszernek tekinthetjük.

Néhány cég nehezményezte, hogy rengeteg plusz munkát kellett végezniük ahhoz, hogy régebbi Windows alkalmazásaikat áttegyék OS/2 Presentation Manager alá. Ez már a múlté. Mindazok, akik figyelembe veszik a Windows tervezési specifikációit, biztonságban érezhetik magukat. Ez természetesen nem vonatkozik azokra a szoftverházakra, ahol megírják az OS/2 Presentation Manager alkalmazásoakt, és nem tesztelik le ezeket a Windows könyvtárral. Bill Gates hisz az OS/2 Presentation Manager jövőjében, cége több OS/2 szoftvert szállított, mint bárki más. (Communications Manager, LAN Manager, Microsoft Word és Excel a Presentation Manager-hez)

Igen rossz véleménnyel van ezzel szemben a Unix operációs rendszerről. Szerinte ugyanis öt nagy céget sem lehet



találni, amely ugyanazt a Unix verziót használná. Vannak olyan Unix variánsok, mint például a NeXT és a Sun, amelyekben egyetlen sornyi kód sem közös. Az OS/2 alapvető célja az, hogy az egyes gépek között az alkalmazások átvihetőek legyenek. A hihetetlenül szerteágazó grafikus interfészek és kernel változatok sokasága elképzelhetetlenné teszi, hogy a különböző gépek Unix alkalmazásai valaha is hordozhatóvá válhassanak. A nyílt rendszerekkel kapcsolatos Gates vélemény is igen egyéni. Szerinte a lehető legnyitottab a PC. Mindenképpen nyíltabbnak tekinthető, mint egy olyan rendszer, ahol a billentyűzet, az interfész és a kód nem szabványos. A Unix csak hivalkodik azzal, hogy ő a nyílt rendszer. Az európai kormányoknak azonban, ha igazi nyílt rendszereket akarnak, a legnyitottabb rendszer, a PC felé érdemes fordulniuk.

A Microsoft csakis azért dolgozott együtt az SCO és a DEC rendszereivel, mert úgy gondolta, hogy mindenkinek joga van olyan konfigurációt választani, ahol buta terminálok kapcsolódnak a server géphez. A LAN Manager-t olyanra fejlesztették, hogy bármelyik operációs rendszerrel képes dolgozni. A DEC céggel való kapcsolat pedig arra szolgált, hogy a nagyobb, régebbi gépek területére is betörjenek. Azoknak a cégeknek, amelyeknél csak többfelhasználós rendszerre van szükségük, annak a Unix megoldást jelent. Azok azonban, akik több felhasználóval, a LAN lehetőségek felső szintjén kívánnak dolgozni, csakis az OS/2-ben gondolkodhatnak.

(Personal Computer Word, 1991/május)

A csúcsgép

Már sok íróasztalon ott található a 25 MHz-es, 80386-os gép. Alig néhány évvel ezelőtt ennek a sebessége, teljesítőképessége még elképzelhetetlen volt. Csaknem minden feladatunkat megoldja néhány másodperc alatt. Igen ám, de az emberi állandóan elégedetlen a meglévővel. Számoljunk csak! Legfeljebb 2 milliárd másodperc van még hátra életünkből. Hát nem akarunk ebből milliókat arra pazarolni, hogy egy-egy Windows alkalmazás képernyője aktualizálódjék, vagy várni, amíg a hosszú programok végre betöltődnek, nem is szólva az „Out of Memory” és hasonló ronda üzenetek bámulásáról. Képzeljük hát el, milyen lehetne a legszuperebb gép, csúcs CPU-val és lemezmeghajtóval, a legjobb monitorral, video adapterrel, egérrel, modemmel, és persze csúcsszoftverekkel. Egy kicsit azonban visszamegyünk a Földre. Mindezt álmodjuk úgy össze, hogy megnézzük a vezető cégek kínálatát az egyes alkotórészekre. Figyelem! Ha már egyszer álmodunk, álmodjunk nagyot. Minden egyes alkotóelemből a legeslegjobbat próbáljuk kiválasztani. Ne elégedjünk meg a kompromisszumokkal!

Gondolkozzunk el a dobozáról. Magát a dobozt hívták korábban CPU-nak, még mielőtt a cserélhető CPU kártyák megjelentek volna. A legkevesebb, amit kívánhatunk, egy 33 MHz-es 80486-os processzor. Igaz ugyan, hogy a 386-os chip megközelítőleg ugyanazt a teljesítményt nyújtaná, de a jövőt is figyelembe kell vennünk. Mivel biztosan akarunk a csodagéphez beépített matematikai koprocesszort és 8 K cache-t, nem ragadhatunk le a 386-osnál.

Mindenképpen rengeteg memóriát szeretnénk, mégpedig úgy, hogy ne kelljen erre bővítő dugaszt elpazarolnunk. A jobb 386-os és 486-os gépekben már 64 bites adatút köti össze a RAM-ot és a processzort, megvan az úgynevezett „interleaving” vagy „bank switching” technika. Ez az „interleaving” két memória chip sorozatot kezel, így a CPU az egyik részből kiveheti az adatokat miközben a másik részt frissíti.

A jobb PC-k már tartalmaznak 8 K cache-t. Ennek az a funkciója, hogy amikor a processzor adatot kér a RAM-ból, ne kelljen minden alkalommal feleslegesen várakoznia. Olvasáskor ugyanis a következő olvasás várható adatait is kiveszi és beteszi a cache-be. És legtöbbször igaza van, tényleg azokat az adatokat fogja kérni legközelebb. A „bank switching” és a RAM cache együtt azt eredményezi, hogy a mikroprocesszor egyszerre 32 bit adathoz jut, méghozzá várakozási idő nélkül.

Biztos, hogy minden 486-os gép, amelynél megvan a RAM cache és az „interleaving” technika, már jó eséllyel indul versenyünkben. A doboz mindenesetre legyen nagy. Még nem is tudjuk pontosan, mi mendent akarunk beleerőszakolni. Egyetlen meghajtó nyilvánvalóan nem lesz elég. A csúcs PC-be legalább kettő, de inkább három kell. Minél több, annál jobb.

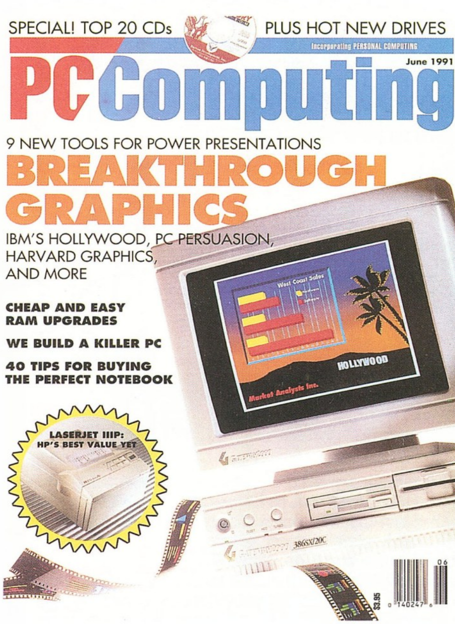
Ugyanez igaz a bővítő helyekre is. Utolsó pillanatban még valami csodát akarnánk hozzátenni, és nincs hová. Ez igazán rémes lenne.

A bővítő dugaszok kérdése újabb problémát vet fel. Milyen legyen: ISA, Micro Channel vagy EISA? Legtöbbször elég lenne az ISA. (ISA = Industry Standard Architecture = AT gépekhez tervezett 16 bites adatsín.) A ma kapható adapter kártyák többsége jó az ISA adatsínhez. Ne kössük meg azonban a saját kezünket! Hátha kell majd a 32 bites kártyakapacitás valamikor. Marad tehát az IBM Micro Channel-je és az EISA (EISA = Extended Industry Standard Architecture).

A Micro Channel mellett szól az, hogy jó az XGA (Extended Graphics Adapter) video kártyához, ami tényleg szuper. Ha ez az XGA szabvánnyá válik, akkor azonban biztosan kihozzák más gyártók ennek ISA és EISA verzióit. Legyen tehát inkább az EISA, ez ugyanis kompatibilis mind az ISA, mind pedig a 32 bites kártyákkal!

Kell tehát az eddigiek alapján egy 33 MHz-es 486-os nagy teljesítménnyel, nagy dobozzal, sok bővítő hellyel, sok meghajtóval, RAM cache-sel. Több tucat PC tudná ezt nyújtani, de hosszas háttérinformáció gyűjtögetés után a táblázatban felsoroltak maradtak versenyben.

Rendszer	Proc. sebesség	Meghajtó helyek	Max. RAM (MB)	Bővítő dugasz EISA	ISA	XT
ALR PowerCache 33/4e	72.1	7	128	6	1	1
Compaq 486/33L	70.5	5	100	5		
Everex STEP Megacube	71.6	8	64	10		1
Gateway 486/33	71.3	5	64	8		
HP Vectra 486/33T	71	6	64	7		
Tandon Tower 486	69.4	8	64	6	2	
Zeos 486/33	70.4	8	32	8		



A táblázatban szereplő processzorsebesség a Norton System Index benchmark szerinti érték. Ez azt jelenti, hogy az összes kiválasztott típus mintegy 70-szer gyorsabb, mint az eredeti 8088-as PC, és több mint kétszer olyan gyors, mint egy 33 MHz-es, 386-os gép. Ennek alpján ugye nem érdemes egyik javára sem döntenünk. Mindegyik típus tartalmaz cache-t, és valamilyen interleaving technikát.

A felsoroltak közül először a Compaq és a Gateway esik ki. Nincs bennük elég meghajtónak hely, ha szupergépben gondolkodunk. Hova tegyük esetleg a CD-ROM-ot vagy a mágneses-optikai meghajtót? A HP Vectra meggyőző kiállású, masszív dobozával. Áramköri tervei mestermunkák. Mivel „file server"-nek tervezték, annyi a bővítési lehetősége, amennyit épeszű felhasználó már nem is kívánna. De ki mondta, hogy most épeszűen gondolkozzunk? Arról volt szó, hogy csodát álmodunk. Tehát nem kell a HP, van olyan gép, ahol még több a bővítőhely. Kiesik a Zeos is, a 3 1/2"-es meghajtóival.

Az ALR sebessége a legnagyobb, a Tandon viszont több bővítési lehetőséget nyújt.

Miért válasszuk mégis az Everex STEP Megacube-ot? Nyolc meghajtója és a 10 EISA slot miatt. Ennyi egyetlen más gépben sincs. 256 K-s a külső RAM cache, 400 Wattos, ezzel még Frankensteint is életre kelthetnénk! Ezek a jellemzők már önmagukban is elég meggyőzőek lennének az Everex választásához. Megjelenése azonban legalább ilyen kiváló. Tényeleg kocka alakú (neve — Megacube, cube=kocka), masszív, csavarhúzó nélkül eltávolíthatók az oldalsó panelek mindkét oldalon, ha bővítő kártyát akarunk betenni.

Megvan az alapgép, legközelebb felvisszük a maximális 64 K-ra a RAM-ját!

(PC Computing, 1991/június)

Szerződéseket olvasva

Volt idő, amikor az emberek tényleg elolvasták a szoftverfelhasználói szerződéseket. (Valamikor talán még a kézikönyveket is, de most nem erről van szó.) A szoftverfelhasználói szerződésektől vagy nevethetnékünk támad, vagy elakad a lélegzetünk. Nevetést vált ki a felesleges ismétlésektől hemzsegő, elképesztő jogi szöveg, lélegzetünk pedig a bosszantással felérő feltételektől akad el.

A szoftverkalózkodásról ejtett ejnye-bejnye mellett a szerződésekben olyasmit is olvashatunk, hogy az engedély nélküli másolatok használata tönkreteheti a merevlemezt és egyéb bajokat okozhat. Idővel azonban a szerződések szövegét annyira uniformizálták, hogy senki nem olvassa el azok szövegét, és a jogászok az „Olvassa el az alábbi fontos információt! felirat helyére legszívesebben olyasmit írnának, hogy „Kenőpénz: lásd az alábbiakban!" Így aztán kész csoda, ha észrevesszük az újításokat, mint például a Samna cég Amí Professional lemezeire vonatkozó üzenetet: „Az Amí Professional szoftvercsomagjához mellékeltük mind az 5,25"-ös, mind a 3,5"-es lemezkészletet. Ön azonban csak az egyiket bonthatja fel, a másikat vissza kell küldenie postán a mellékelt dobozban a Samna céghez." Hogyan? Megvesz az ember egy dobozt tele mindenféle anyaggal, és az nem is az övé? A hátoldalon azután megmagyarázzák, hogy „az én kényelmem érdekében" van kétféle lemez mellékelve, s abból csak azt nyithatom ki és használhatom, amelyik az én gépemhez való, a másikat harmadik félnek semmilyen módon nem továbbíthatom stb. Na, és mit csinálnak, ha megtartom a lemezeket? A vevőket eleve tolvajként kezelni általában nem jó ötlet.

Körülnéztem egy kicsit a többi szoftvernél. A WordPerfect egyszerűen azt mondja, hogy egyetlen gépéhez engedélyezi egy lemezkészlet használatát. Ha rá akarod tenni a hordozható laptop-odra, vedd meg még egyszer! A Microsoft engedékenyebbnek látszik, de megfeneklik a „ha" feltételek tengerén. Ha a szoftvert 80%-ban a bejegyzett számítógépeden használod, akkor ráteheted még a hordozható vagy az otthoni gépedre. De mi van akkor, ha a hordozható gép a „bejegyzett"?

A Borland világosabban beszél: „Ezt a szoftvert úgy kell kezelni, mint egy könyvet." — azaz átviheted máshova, de mindig csak egy helyen szabad használni. Ha a laptop-oddal elutazol egy másik városba, és az irodai géped elzárható, akkkor minden rendben. De ha nem, akkor valaki használhatja közben azt a programot! Lehet, hogy senki nem használja, de elvben használhatja. Ez akkor vajon szerződésszegés?

Csak a legdörzsöltebb ügyvédek tudhatják!

(PC Computing, 1991/július)

TERMÉSZETESEN SPÓROLJON

Festékkazetták és fotóhengerek felújítása



Fénymásoló és lézernyomtató-gépek használói figyelem!

Festékkazetták, fotóhengerek és egyéb alkatrészek felújítását elvégezzük. Mindezt különböző típusú fénymásolókhoz és lézernyomtatókhoz svájci-japán technológiával és alkatrészekkel.

Jellemzői:
- Megbízható.
- Kitűnő minőségű.
- Az új kazettánál 50%-kal olcsóbb.
- Kevesebb a hulladék, így környezetkímélő.

FARBAX®
Refill-Service

Farbax Kft.
Márkabolt és Ker. Iroda:
H-1138 Budapest, Váci út 178.
Tel.: 120-8053, Tel./Fax: 120-1241
Refill Service:
H-9700 Szombathely,
Rózsa F. krt. 2.
Tel.: 94-16839, Fax: 94-13868

Info Coupon:

☐ Kérjük részletes tájékoztatójukat

☐ Kérjük szakértőjük személyes látogatását

Név: ____________

Cégneve: ____________

Részleg: ____________

Cím: ____________

____

____

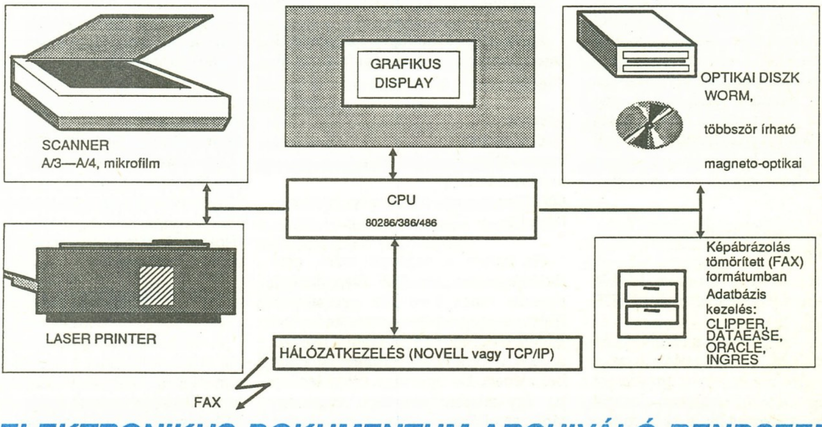
INFORMÁCIÓKÉRÉS: 30 ▼

VT-SOFT®
VIDEOTON SOFTWARE KFT

1033 Budapest, Vörösvári út 103–105.
Telefon: 180-3750, 180-3744
Telex: 22-6192 Telefax: 180-3750

ÚJ!

INTEGRÁLT KÉP- ÉS SZÖVEG-FELDOLGOZÁS



ELEKTRONIKUS DOKUMENTUM ARCHIVÁLÓ RENDSZER

Kulcsrakész alkalmazások az egyfelhasználástól a nagy kapacitású LAN rendszerekig.

INFORMÁCIÓKÉRÉS: 20 ▼

Framework-világ

Vezérlésátadó függvények

Cikksorozatunk első részében az egyszerű programok készítéséig jutottunk el a Fred nyelvű programozásban. Ismertettük a programok készítésének alapvető szabályait, példákkal illusztráltuk az adatbevitelt és az eredmények képernyőre íratását. Ezúttal a vezérlésátadások különböző módjait tárgyaljuk. Emlékeztetőül még annyit: az adatbevitel és a kiíratás olyan belső függvények eredménye, amelyek karakteres adatokkal (stringekkel) dolgoznak. Ezért kell nagyon odafigyelnünk a konvertáló függvények használatára! A változók alkalmazása esetleg az eredmény kiíratásakor igényel átalakítást (stringgé), ezért ilyenkor a kiírandó teljes karakterlánc összeállítása az & operátor segítségével történik.

Rátérve mostani témánkra, előrebocsátjuk, hogy az e részt kiegészítő, illusztráló mintaprogramokat — az esetleges sajtóhibákat kiküszöbölendő — a lemezmelléklet \Fred könyvtárában helyeztük el. Ne feledjük: a programot tartalmazó keret neve a leírásban az első sorban olvasható! (A keret fájlneve ennek első 8 karaktere, ami helyett — ha az eredeti karakter DOS-ban le van tiltva — az _ jelet találjuk. A kiterjesztés mindig .FW3.) A programok indításáról is volt már szó: a kurzorral a programot tartalmazó keret szegélyére kell állni, majd F5!

Feltételes vezérlésátadás: @IF(a,b,c)

Ez a függvény a program kétfelé ágaztatására nyújt lehetőséget (TRUE ág, FALSE ág).

A következő program a billentyűzetről beírt számról eldönti, páros-e vagy páratlan. Az eredmény a programot tartalmazó keret belsejében jelenik meg: más az üzenet, ha a szám páros volt, s más, ha páratlan. Ezt az @IF függvény alkalmazásával éri el.

```
;páros-páratlan
@local(szam),
szam:=@value(@inputline("Írd be a
számot!")),
@eraseprompt,
@if(@mod(szam,2)=0,
   @prompt("Páros számot írt be!"),
   @prompt("Páratlan számot írt be!"))
```

A második példaprogram a billentyűzeten begépelt telefonszám hosszát ellenőrzi.

```
;joszam
@local(szam),
szam:=@inputline("Írd be a
telefonszámot!"),
@if(@len(szam)<>7,"A szám hossza
hibás!")
```

Ha helyes a begépelt szám, nem történik semmi; az @IF függvénynek ugyanis nincs harmadik paramétere. Ebben a programban az eredmény ott jelenik meg, ahová a programot beírtuk: ezúttal a programot tartalmazó keret belsejében. De ugyanígy megjelenhetne egy adatbázismezőben vagy egy táblázatcellában is.

Mindkét programban az @IF függvényben csak egy üzenetet kellett összeállítani, a feltétel teljesülésétől függően. Ha azonban bármely (TRUE vagy FALSE) ágban több kifejezés végrehajtása szükséges, akkor ezeket egy @LIST függvény argumentumaiként kell elhelyezni, az @IF ugyanis csak három, kötött jelentésű paramétert tartalmazhat.

Nézzük meg a @LIST alkalmazására a következő példát!

```
;joszam1
@local(szam),
szam:=@inputline("Írd be a
telefonszámot!"),
@if(@len(szam)<>7,
  @list(@beep,
  "A szám hossza hibás!"))
```

Hibás adathossz esetén nemcsak hibaüzenetet, hanem sípjelet is kapunk.

Az @IF függvények láncolására szolgál a következő példaprogram:

```
;szam
@local(szam),
szam:=@value(@inputline("Írd be a
számot!")),
@if(szam=0, "A begépelt szám 0",
  @if(szam>0,
"Pozitív számot gépeltél be!",
"Negatív számot gépeltél be!"))
```

Többszörös feltétel esetén az @IF-ben a logikai függvényeket (@or, @and) is használhatjuk:

```
;es
@local(szam),
szam:=@value(@inputline("Írj be
  egy számot 5 és 10 között!")),
@if(@and(szam>=5, szam<=10),
"Jó! "& @integer(szam) & " 5 és 10
között van!",
"Rossz! "& @integer(szam) &" nem 5
és 10 között van!")
```

A Fred ciklusfüggvénye (@while(feltétel, kif.,...))

A **while** függvény alkalmazása esetén mindaddig, amíg a feltétel eredménye #TRUE, végrehajtódnak a feltételt követő kifejezések (a ciklus magja). Mihelyst a feltétel hamissá válik, a program kilép a @while függvényből, ezután már nem hajtódnak végre többé a ciklusmag kifejezései.

Következő példánkban a ciklusban mindaddig **növekszik a szamlalo** értéke, amíg el nem éri az 50-et.

```
;ciklus1növ
@local(szamlalo),
szamlalo:=0, ;felesleges: ez az
alapértelmezés
@while(szamlalo<50,
  szamlalo:=szamlalo+1),
ciklus1:=szamlalo
```

A program utolsó sora adja meg a program értékét, az értékadás művelettel ezt hangsúlyoztuk.

Ugyanez a program — a számláló értékét 0-ig **csökkentve** — tanulmányozható az alábbi listán:

```
;ciklus1csökk
@local(szamlalo),
szamlalo:=50,
@while(szamlalo>0,
  szamlalo:=szamlalo-1 ),
ciklus1csökk:=szamlalo
```

Gyakran szükséges, hogy a program mindaddig kérje az adatot, amíg nem érkezik helyes adat. Ha például hónapsorszámot várunk, a szám nem lehet 1-nél kisebb és 12-nél nagyobb. Amíg a beírt szám nem felel meg ennek, a program újra kéri a számot:

```
;ciklus3
@local(szam),
szam:=0,
@while(@or(szam>12,szam<1),
szam:=@value(@inputline("Írd be
  a hónap sorszámát!"))),
"A hónap sorszáma "&@integer(szam)
```

A következő példában egy logikai változót használunk a ciklus folytatásának eldöntéséhez. Ennek értékét — ha szükséges — a ciklusmagban változtatjuk meg, s így a változtatást követő vizsgálat után lépünk ki a ciklusból.

```
;ciklus4
@local(szamlalo, tovabb),
szamlalo:=@value(@inputline("Add
meg a számláló kezdőértékét!")),
tovabb:=#TRUE,
@while(tovabb,
  ciklus4:=szamlalo,
  szamlalo:=szamlalo+1,
  @if(szamlalo>=50,
    tovabb:=#FALSE )),
ciklus4
```

A program eredményül azt adja meg, hogy ténylegesen hányszor került végrehajtásra a ciklusmag.

Elágazás többfelé (@select)

Ha egy programban a változó értékétől függően nem kétféle, hanem többféle válasz (művelet) lehetséges, ez megoldható @IF függvényekkel, de rövidebb, áttekinthetőbb, ha például a **@select** függvényt használjuk.

Példa: a billentyűzetről számot várunk, amely 1 és 4 közé esik. A szám nevét kell eredményként előállítani.

```
;select1
@local(szam),
szam:=@value(@inputline("Írj be egy
számot 1 és 4 között!")),
"Ez a szám:" & @select(szam, "egy",
"kettő", "három", "négy")
```

Ha nem az 1—4 számok közül ütünk be egyet, akkor hibaüzenetet kapunk (illetve ha 0-t, akkor csonka eredménystringet).

Ezt a hibát oldja fel a következő program: a programból csak úgy tudunk kilépni, ha helyes adatot (1—4 közötti számot) adunk meg.

```
;select2
@local(szam),
szam:=0,
@while(@or(szam<1,szam>4),
szam:=@value(@inputline("Írj be egy
számot 1 és 4 között!"))),
"A beírt szám: " & @select(szam,
"egy",
"kettő", "három", "négy")
```

Újabb példánk egy programmakró. A makró meghívásakor magyarul kiírja a képernyő alján, hogy az aktuális gépi dátum a hét melyik napjára esik. Ezt a programot a NAPMAKRO keret képletterületére írt @setmacro függvénnyel definiáljuk makróként. Maga a makródefiníció e keretben levő N nevű kereten található.

```
;napmakro
@setmacro({alt-n},napmakro.n)
```

A kifejezés értelmében az **alt-n** billentyűkombináció leütésére történik a makródefiníció végrehajtása.

```
;napmakro.n — a makró definíciója
@local(n,nap),
n:=@mod(@diffdate(@today,@date(199
1,1,1)),7),
@if(n<0,n:=n+7),
nap:=@select(n+1, "kedd", "szerda",
  "csütörtök", "péntek", "szombat",
  "vasárnap", "hétfő"),
@eraseprompt,
@prompt(nap)
```

Van a Fred-ben egy másik, a @select-hez hasonló funkciójú függvény is: **@choose**. Paraméterei, használata a @select-ével ebben az esetben megegyeznek.

Kilépés a formulából (@return és @result)

A Fred nyelvű programból háromféleképpen lehet kilépni: explicit módon a @return és a @result függvénnyel; implicit kilépés akkor történik, ha végrehajtódott a formula utolsó sora.

Mindkét kilépési függvény csak az aktuális formulából enged kilépni; ha ez egy szubrutin volt, a vezérlés visszajut a hívó programra.

A két függvény között csupán az a különbség, hogy a @return által visszaadott érték a #FUNCTION konstans, a @result függvényé a megadott paraméterérték, amely a keret belsejében megjelenik. Ezt a paramétert feltétlenül meg kell adni, ha mást nem, hát például 0-t argumentumként.

Erre is bemutatunk egy példát.

```
;hivo
szubrutin:=" ",
hivo:="Szia",
hivo:=@szubrutin
;szubrutin
@local(szam),
@while(szam<100,
  szam:=szam+1 ),
@result(szam),
szubrutin:="vége"
```

Ha elindítjuk a HIVO-t, az meghívja a SZUBRUTIN-t (lásd később), amelynek @result függvénye kilépést eredményez a SZUBRUTIN-ból 100 értékkel, s ezt az értéket veszi fel a HIVO is az utolsó értékadó kifejezés miatt. Így sohasem kapjuk eredményül a **vége** stringet.

Ha a @result függvényt felcseréljük @return-nel, a HIVO értéke 100, a SZUBRUTIN-é viszont #FUNCTION lesz.

Megfigyelhető, hogy hiányzik a Fred függvénykészletéből a GO TO típusú elágazási lehetőség.

Program — modulokból

A programhívások szubrutinhívásnak felelnek meg: a hívott program végrehajtása után a vezérlés visszaadódik a hívó programnak, a programhívó kifejezést követő kifejezés végrehajtásával. Az előző példa ezt is illusztrálja.

A programokat egyébként is célszerű modulokra bontani, amelyek külön-külön megírhatók és tesztelhetők. A modulok egyszerűen beépíthetők a programba, felhasználhatók más programokban is.

A modulok hívása a programban (szubrutinhívás) — **@programnév** — ugyanolyan, mint ha egy paraméter nélküli függvényt hívnánk.

Mivel ilyenkor a memóriában kell lennie minden olyan keretnek, amely a programban használt szubrutint tartalmazza, célszerű ezeket egy konténerkeretben elhelyezni. Fontos azonban, hogy ha egy üres keret képletterületén levő programot már futtattunk, nem helyezhetünk el benne újabb kereteket, mivel az ilyenkor értékkeretté válik.

Egy keret tartalmazhat a képletterületén programot, de ugyanaz a keret használható globális változóként is.

```
;főprogram
@bevitel,
@hossz,
@valasz
```

továbbá

```
;bevitel
@inputline("Írd be a neved!")
;hossz
@len(bevitel)
;valasz
@eraseprompt,
@prompt("A "& bevitel &" "&
@integer(hossz)&
" betűből áll",(80-hossz)/2)
```

A fenti példában a **főprogram** @BEVITEL kifejezésének hatására végrehajtódik a BEVITEL kereten lévő program, majd a hívó kifejezést követő lépés hajtódik végre; a @len(BEVITEL) kifejezés a BEVITEL nevű (string típusú) globális változó (a keret belsejében található) aktuális értékének a hosszát határozza meg. Végül az eredmény kiíratása történik meg a @VALASZ kifejezés hatására.

Vegyük észre, hogy funkcióját tekintve ez ugyanaz a program, amely az előző számban szerepelt NEVHOSSZ néven, csak most másképp írtuk meg.

Az utolsó példánk programjait tartalmazó kereteket tegyük egy üres keretbe, hogy a futtatáshoz mindig valamennyi a memóriában legyen. A befoglaló konténerkeret képletterületére kell egy olyan kifejezést írni, amely átadja a vezérlést a FŐPROGRAM-nak: **@program.főprogram**. Ennek indítása után a programunk kifejezései rendre végrehajtódnak.

Ha akár globális változó használatakor, akár programhíváskor egy **keretre hivatkozunk**, nem mindig elegendő a keret nevének a megadása, hanem kell a hozzá vezető útvonal is. Ennek szabályai a FRAMEWORK interaktív használatából már ismertek: elválasztó karaterként pontot teszünk, s a keretnevek megadásánál kívülről befelé haladunk, ahogy a példában is láttuk.

A **szülőkeret** mindig konténerkeret. Azok, amelyek ebben vannak, a **gyerekkeretek**. **Testvérkeretek** azok, amelyeknek ugyanaz a szülőjük. Nem kell megadni az útvonalat, ha a hívó és a hívott keret ugyanazon a szinten van, és akkor sem, ha szülőkeretre hivatkozunk. Ha azonban gyerekkeretről van szó, meg kell adni az útvonalat.

A következő részben a billentyűzet programozásáról lesz szó; a billentyűleütéseket kezelő függvényeket ismertetjük.

Gyakorló feladatok

1. Készítsen olyan programot, amely a billentyűzetről bekért adatokból — év, hónap (számmal), nap — a képernyő alján kiírja a dátumot az alábbi formában: 1991. július 12.

2. Készítsen programot egy gyerek szorzótábla-ismeretének ellenőrzésére! A program először kérje be, milyen tartományban várjuk az összeszorzandó számokat, majd generáljon két számot a megadott tartományban. Ha a feladat eredményét helyesen adja meg, a program dicsérje meg és bízza választására, kér-e új feladatot. Ha a válasz hibás, kérjen a program új tippet. Ha ez jó, adjon új feladatot, ha nem, dorgálja meg a gyereket, és adja meg a helyes eredményt, majd természetesen újabb feladatot. (Vagyis a programból csak úgy lehet kilépni, ha a válasz elsőre helyes).

A múlt havi gyakorló feladatok megoldása

1.

```
;{Alt-F5}
@eraseprompt,
@local(szoveg),
szoveg:="Önnek még " &
@integer(@memavail) &
"bájt szabad RAM-terület áll
rendelkezésére",
@prompt(szoveg,(80-@len(szoveg))/2)
```

A lemezmellékleten F5 néven található az a konténerkeret, amelyet F5-tel indítva alt-m-mel hívható az új makró. Tetszés szerint azonban a belsejében levő keretet a Library Macros szekciójába is másolhatjuk, s ha a nevét **{alt-F5}**-re módosítjuk, s egyúttal töröljük a régi alt-F5 keretet, a továbbiakban az újat használhatjuk a régi makródefiníció helyett.

2.

```
;Szorzás
@local(a,b,c,d,tizedes),
a:=@value(@inputline("Add meg a
szorzandó értékét, majd üss
ENTER-t!")),
b:=@value(@inputline("Add meg a
szorzó értékét,    majd üss
ENTER-t!")),
c:=a*b,
tizedes:=@value(@inputline("Az
eredmény kért pontossága
(tizedesjegyek száma)?")),
@eraseprompt,
d:=@decimal(a,tizedes)&" *
"&@decimal(b,tizedes)& " = "&
@decimal(c,tizedes),
@prompt(d,(80-@len(d))/2)
```

A programban — bár ez nem szerepelt a feladatban — lehetőséget adtunk törtrészt tartalmazó számok szorzására is. Az adatokhoz illeszkedő pontosságú eredmény képzéséhez a szükséges tizedesjegyek számát is kérjük, amit a kiíráshoz használt @decimal konvertáló függvényben adunk meg a **tizedes** lokális változóban.

A lemezen található ADATELLE.FW3 fájl a feladatot ellenőrzött adatbevitellel oldja meg: továbblépés előtt visszakérdez: helyes-e a beírt adat. Ha nem, javítható.

Kóczy A. Judit

Konferencia a szoftver jogvédelméről

A Neumann János Számítógéptudományi Társaság, a Központi Statisztikai Hivatal és az Artisjus — az Osztrák Számítógéptudományi Társasággal, a Szervezési és Vezetéstudományi Társasággal, valamint a Magyar Iparjogvédelmi Egyesülettel együttműködve — nemzetközi konferenciát szervez

a szoftver szerzői jogvédelméről

1991. október 14—18. között a Hotel Agróban (Budapest XII., Normafa út 54.).
A tapasztalatok azt mutatják — és ezt kívánja tovább erősíteni a nemzetközi konferencia is —, hogy Magyarország úttörő szerepe a szoftver szerzői jogvédelmében változatlanul élénk érdeklődést vált ki nemzetközi téren.
Kiemelkedő nemzetközi szervezetek és előadóik jelezték részvételi szándékukat és küldték meg előadásaikat számos időszerű témáról:
A szoftver szerzői jogvédelme.
Az alkotók személyi és vagyoni jogainak intézményes oltalma.
A hazai és nemzetközi jogalkotás jelenlegi helyzete és várható fejlődése.
A bírósági és államigazgatási joggyakorlat.
A szoftver szerzői jogának és a használati jognak az elhatárolása, jogi és számviteli kérdések.
A magyar személyi jövedelemadózás sajátos (és ellentmondásos) helyzete.
A szoftverdokumentálás és prioritás védelmi hatása és a regisztrálás lehetőségei.
A versenyjog és a szerzői jogvédelem kölcsönhatásai.
A rendezvény nemzetközi program- és szervezőbizottsága:
Fővédnök: Boytha György
A programbizottság elnöke: Havass Miklós.
A szervezőbizottság elnöke: Varga Lajos
Tagjai: Christian Galinski (Ausztria), Gyertyánfy Péter, Emma Nicholson (Anglia), Oberrecht Gabriella, Perjés Sándor, Tóth Istvánné, Martin van Gelderen (Hollandia), Weisz Istvánné, Regina Thaller (Ausztria).
A konferencia szponzorai: a Controll, az IBM Magyarországi Kft. és az IBM következő partnerei:
Égszi-Hardsoft, Égszi Rendszerház, Ganz-IAS, H&H, InterCode, Intersoft, Kerszi, Mutex, Polygon, Rolitron-Gyöngyös, Számalk-Softex, Volán Elektronika.
Az elfogadott előadások a következők:
1. Számítógépi programok védelme és a szerzői jog fejlődése a világban (Boytha György).
2. Algoritmusok, nyelvek és interfészek jogvédelme, az információtechnológia recessziójának kezdete? (C. H. Cap, Svájc).
3. Szoftverjogvédelem az európai közösségi jog keretében (P. L'Ecluse, Belgium).
4. Szoftverlicenc-szerződések (Eckhardt Péter).
5. A szoftvervédelem módszerei (Éltető László).
6. Jogi alapok a nemzetközi szoftverkereskedelem könnyítésére (R. N. Freed, USA).
7. A szoftver szerzői jogi védelme az Európai Gazdasági Közösség országaiban (A. Hellfeld, Németország).
8. A jelenlegi európai közösségi irányelv hatása a szoftverjogvédelemre (R. Innis, Anglia).
9. Szükség van-e Ausztriában a továbbiakban az ACM-re vagy a kiadók nagyobb felelősségére a szoftver fokozottabb jogvédelme érdekében? (J. A. Kaltner, Ausztria)
10. Munkavállaló szerzők által létrehozott piacképes szoftver fejlesztése (A. Kindermann, Németország).
11. Az utánzás a hízelgés legőszintébb formája? (I. Lloyd-M. Simpson, Anglia)
12. A szoftver szerzői jogvédelmének aktuális fejlődési irányai a Cseh és Szlovák Köztársaságban (Z. Loebl, Cseh és Szlovák Köztársaság).
13. A szoftver szerzői jogi védelme egy gyakorlati szakember szemszögéből (M. Macdonald, Anglia).
14. Szoftvervisszafejtés a szerzői jog megsértése nélkül (V. Marsland, Anglia).
15. Szoftver szerzői jogi védelme az Egyesült Királyságban (E. Nicholson, Anglia).
16. A szoftver szerzői jogi védelmének gyakorlati kérdései a magyar piacon (Oberrecht Gabriella).
17. A szoftverjogvédelem gyakorlati problémái (Pálos György).
18. A szoftverszerzői díjak adózása Magyarországon (Perjés Sándor).
19. A megbízásra készített szoftverprogramok — konfliktus a szabadalommal (J. Phillips, Anglia).
20. A szoftver szerzői jogi védelme Jugoszláviában (C. Popovic, Jugoszlávia).
21. A szoftver jogi védelme Kelet-Európában: hozzájárul-e a jobb kelet—nyugati kereskedelmi kapcsolatokhoz (C. Prins, Hollandia).
22. Számítógépes programok szerzői jogi védelme: a törvény fejlődése az USA-ban (V. Siber, USA).
23. A szoftver szerzői jogi védelme Dél-Afrikában (H. Staniland, Dél-Afrika).
24. Az Európai Közösség szoftverirányelve és ennek megjelenése a kölcsönös szoftverfejlesztésekben (T. Vinje, Belgium).
25. Hogyan hathat a számítógépes bűnözés regisztrálása a szoftver szerzői jogi védelmére Európában? (I. Walden, Anglia)
26. Szoftverjogvédelmi törvény a fejlődő országokban (N. J. Wilkof, Izrael).
A részvételi lehetőségekről a Neumann János Számítógéptudományi Társaság titkársága (Budapest V., Báthori u. 16., telefon: 1329-349, 1329-390) szolgál felvilágosítással.

INFORMÁCIÓKÉRÉS: 32 ▼

Unix-konferencia és -kiállítás Magyarországon

A 70-es évek elején fejlesztették ki a Unix „prototípusát", amely — korát kissé megelőzve — a gépfüggetlenség és a portabilitás eszméjére épült. Ezt a többfeladatos és többfelhasználós operációs rendszert hamar megszerették mind a programozók, mind pedig a kutatók. A Unix valódi térhódítása azonban csak a nagy teljesítményű mini- és mikroszámítógépek megjelenésével kezdődhetett meg. Számos cég dolgozta ki saját implementációját, illetve készítette és tervezi a termékeit a Unix operációs rendszer alá. A Unix a nyílt rendszerek alapkövének tekinthető, napjaink egyik legelterjedtebb operációs rendszere.
Az európai Unix-felhasználók szervezete (EurOpen) 1989-ben vette fel tagjai közé hivatalos kelet-európai tagként a Magyarországi Unix Felhasználók Csoportját (HUUG). Jelenleg a HUUG-nek 21 tagja van. A tagfelvétel időpontja azért is különösen jelentős, mert abban az időben a Unix még szerepelt a COCOM-listán.
Az EurOpen nagy segítséget jelent a Unix-felhasználóknak, ugyanis a negyedévenként megjelenő hírlevelekből a legfrissebb információkhoz juthatnak a tagok, valamint csatlakozhatnak az EuNet Unix-hálózatra is. A tagszervezetek munkáját az EurOpen időszakos kiadványokkal és tanfolyamokkal is segíti. A „kebelbeliek" a Unix-fejlesztések és -alkalmazások legújabb eredményeiről az évente két alkalommal, ősszel és tavasszal megrendezett konferenciákon és a hozzájuk kapcsolódó kiállításokon kaphatnak tájékoztatást.
Ebben az évben az EurOpen szokásos őszi rendezvényének színhelyéül Budapestet választotta, amely a hazai szakma elismerését jelenti , hiszen Európa legnagyobb méretű Unix-konferenciája és -kiállítása először lesz egy „volt szocialista" országban. Ezzel a rendezvénnyel szeretné az EurOpen elősegíteni a Unix még gyorsabb hazai elterjedését. A konferenciával egy időben megtekinthető kiállítás jó alkalmat teremt a hazai cégek bemutatkozására, valamint nemzetközi megmérettetésére.
A Unix-konferencia időpontja: 1991. szeptember 18—20., a kiállításé pedig: 1991. szeptember 16—20. Mindkettőnek a Budapest Kongresszusi Központ ad otthont.

INFORMÁCIÓKÉRÉS: 33 ▼

Belépőjegy: az Assembly

Kéj és kín — egyszerre

Az előző rész végén azzal búcsúztam, hogy az Assembly-programozás könnyen felejthető kaland lesz. Erre a megjegyzésemre talán sok szakmabeli összeráncolta a homlokát, de mivel a sorozat elsősorban nem nekik szól, csak azt tudom mondani: az Assemblyről szóló részt kizárólag azzal a céllal illesztettem a sorozatba, hogy olvasóim maximálisan értékelni tudják a magas szintű nyelveken történő programozás kényelmeit.

Mint az első részben már szó volt róla, a programozás „hőskorában"(?) a számítógépek programozása kizárólag binárisan történt, úgynevezett vezérlőpultok segítségével. A pult maga kapcsolókból és néhány visszajelző lámpából állt. A (ki-be) kapcsolókkal a számítógép egy-egy áramkörének konkrét fizikai állapotát lehetett beállítani: egy kapcsoló — egy bit alapon. Ugye mondanom sem kell, a címek és adatok bitenkénti megadása nemcsak rettenetesen unalmassá tette a „programozást", de sok hibával is járt. Persze nehogy azt higgye valaki is, hogy a mai mikroprocesszoros gépek a régieknél értelmesebbek. Szó sincs róla: az „egy változóhoz rendeljük hozzá a 12-es értéket" utasítást sajnos a modern gépek sem értik, ők is makacsul ragaszkodnak a 01001111 vagy pontosabban az alacsony szint, magas szint, alacsony szint... formátumhoz, mivel hogy a processzorok által értelmezett utasításformátum mind a mai napig a jó öreg bitminta maradt.

Bitvadászat

A processzor utasításkészlete nem más, mint bináris bemeneti értékek halmaza, amely a processzorban — egy utasításciklus alatt — előre definiált működést vált ki. A processzor működését leíró utasításoknak a processzor adatbemenetein meghatározott ütemezéssel és sorrendben kell rendelkezésre állniuk. A program így olyan utasítások sorozata, amelyek hatására a számítógép egy adott feladatot elvégez. A processzor számára a program nem más, mint bináris számok sorozata. Például:

```
10100000 01100000 00000010
00000010 00000110 01100001
00000010
10100010 01100010 00000010
```

(A bitsorozatot nyomdatechnikai és értelmezhetőségi szempontok miatt bájtonként elkülönítettem.)

Ez az úgynevezett gépi nyelvű vagy tárgyprogram. Sajnos, be kell látnunk: a számítógépek nem programozhatók sem szavak, sem oktális, decimális vagy hexadecimális számok segítségével. A számítógépek kizárólag bináris mintákat képesek utasításként vagy adatként értelmezni. Bárki beláthatja, hogy a fenti kis példaprogramba nagyon könnyen becsúszhat egy hiba, amit nem könnyű észrevenni. Aki nem hiszi, járjon utána:

```
10100000 01100000 00000010
00000011 00000110 01100001
00000010
10100010 01100010 00000010
```

Természetesen a számítógépes szakembereket sem elégítette ki a bináris programozás, és a binárisok helyett igen hamar áttértek az oktális vagy a hexadecimális számok alkalmazására. Persze ahhoz, hogy a számítógép a más számrendszerben írt programokat megértse, szükség volt egyszerű eszközökre, amelyek a konvertálást elvégezték. Az egyik ilyen a hexadecimális billentyűzet volt — ahol a kapcsolókat már billentyűk helyettesítették —, a másik pedig a hexadecimális töltő. A hexadecimális töltő nem más, mint egy program, amely a hexadecimális számokat bináris számokká konvertálja. A program hexadecimális változata rövideb idő alatt előállítható, és ellenőrzése sem olyan fárasztó. A programban elkövetett hiba is könnyebben észrevehető:

```
A0 60 02
02 06 61 02
A2 62 02
```

A hibás kód:

```
A0 60 02
03 06 61 02
A2 62 02
```

Persze azt állítani, hogy a hexakódolás a programot az ember számára jól érthetővé tette, enyhe túlzás. A hexadecimális forma még mindig nehezen olvasható és értelmezhető. Például nincs megkülönböztetve a művelet az adatoktól vagy a címektől, és semmilyen támpont sincs, ami a program funkciójára utalna. Mit jelent mondjuk A0 vagy 02? A processzor utasításait leíró kódtáblázat bemagolása nem valami felvillanyozó ötlet. Ezenkívül a kódok egy másik processzoron teljesen mások lesznek, és az érthetőséghez rengeteg dokumentációt kellene mellékelni.

És lőn Assembly

Egy nyilvánvaló lépést a programozásban az utasításkódokhoz való névrendelés jelentett. Az utasításkód nevét „mnemonik"-nak hívják. Az utasításmnemonik feladata, hogy néhány karakteren leírja: mit csinál az utasítás. Természetesen a szabadverseny törvényeihez igazodva minden processzorgyártó (mivel ők sem képesek a hexakódok megjegyzésére) definiált mnemonik-készletet a saját processzortípusaira. Egy gyártó processzorcsaládjának mnemonikjai általában megegyeznek, de az eltérő típusú processzorokra ez már nem igaz. A gyártó által definiált mnemonikok használata nem kötelező, bár célszerű, mert az adott processzort támogató összes fejlesztő és termék ehhez igazodik.

Az utasítás-mnemonikok az utasításra vagy annak hatására utaló angol szó vagy szavak nem mindig egyértelmű rövidítéséből keletkeztek. Az utasításmnemonikon kívül a gyártók általában a processzor regisztereit is elnevezik. Akárcsak az utasításneveknél, a regiszterek nevei is lehetnek a funkcióra utalók vagy ismeretlen eredetűek, és — akárcsak az utasításoknál — használatuk opcionális, de melegen ajánlott. Bár a hetvenes évek végén történtek próbálkozások az Assembly-mnemonikok szabványosítására, ezek nem vezettek sikerre, így ha valaki munkája során processzort vált, kénytelen többek között az utasításneveket is újra megtanulni. Példaprogramunk az Intel 80x86-os mikroprocesszorcsalád (a PC agya) assembly nyelvén a következőképpen néz ki:

```
MOV AL, [0260H] ADD AL, [0261H]
MOV [0262H], AL
```

Ugyanez Motorola 680x0 processzorra (Macintosh, Amiga):

```
MOVE.B $0260, D0
ADD.B $0261, D0
MOVE.B D0, $0262
```

Itt szeretném leszögezni, hogy az Assembly szigorúan véve nem programozási nyelv, csak a processzor utasításainak és regisztereinek emészthető formában történő megjelenítése. A processzorutasítások két alapvető csoportba sorolhatók: adatmanipulációs és vezérlésszekvenciát módosító utasítások. Az Assembly utasítások nem problémaorientáltak, mint a magas szintű nyelvek alapstruktúrái, „csupán” az adott processzor „nyers” erőforrásainak kezelését teszik lehetővé. A „trükk” persze abban rejlik, hogy ezeknek az egyszerű utasításoknak a megfelelő szervezésével tetszőlegesen bonyolult problémát tudunk a számítógép segítségével megoldani. Az alkalmas kód előállítása persze nem könnyű feladat, és a számítógépes problémák elemzésével bizonyos elemi vezérlési struktúrák ismételt megjelenése, valamint az emberi lustaság és kényelemszeretet felvetette a magasabb szintű programozási nyelvek iránti igényeket. A magas szintű nyelvekről, a magas szintű nyelvek és az Assembly közötti kapcsolatról és a különbségekről a következő részben részletesen is szó lesz, most elég annyi, hogy a magas szintű nyelvek már nem processzor-, hanem problémaorientáltak, és jelölésmódjuk is közelebb áll a beszélt nyelvekhez (legalábbis az angolhoz).

Az assembler

A kérdés az, hogy a számítógép számára most már végképp értelmetlen szöveget hogyan alakítsuk át a számítógép számára érthető bináris kóddá. Az egyik megoldás az Assembly program kézzel történő kódolása. A processzor kódtáblája segítségével az Assembly programot átalakíthatjuk binárís vagy hexadecimális kódolásúvá, és máris futtatható a programunk (az utóbbi esetben a hexatöltő segítségét is igénybe kell vennünk). A kézi kódolás nem valami vidám feladat, unalmas, monoton és persze hibalehetőségekkel teli. (Sorok vagy számok felcserélése, utasítások kifelejtése és a forráskód félreolvasása, hogy csak néhányat említsek az elkövethető hibák közül.)

A legtöbb processzor még azzal is tetézi a problémákat, hogy eltérő hosszúságú utasításokat használ. Egyik utasítás egy bájt hosszú, a másik kettő vagy akár öt, ráadásul az utasításkódot követheti adat-, cím-, regiszterazonosító vagy ki tudja még, mi minden. Ugye mondanom sem kell, ezt a fajta munkát a középkorban gályarabokkal végeztették volna el. Szerencsére „gályarabért” a számítástechnikusnak sem kell messzire mennie, hiszen a számítógép mindenre „rávehető”. Mi sem természetesebb, mint egy olyan program írása, amely az assembly programot (az úgynevezett forrásprogramot) emberi beavatkozás nélkül (ami a mi esetünkben egyben azonos a hibamentességgel is) alakítja át bináris vagy tárgykóddá. A fordítást elvégző program az assembler. Az assembler a kódolás fárasztó munkájától szabadítja meg a programozót, de ennek ára is van. Az assembler használatához már nem elég a processzor utasításkészletének ismerete, mivel az assemblernek magának is vannak bizonyos szabályai, amiket a programírás közben be kell tartani. Ilyen szabályok lehetnek a különböző jelzőknek (szóközöknek, vesszőknek, pontosvesszőknek és kettőspontoknak) a megfelelő helyeken való elhelyezése, az utasítások és regiszterek neveinek helyesírása, a forráskód formátuma stb. Szerencsére ezek a szabályok egyszerűek, és nincs belőlük sok, vagyis könnyen elsajátíthatók.

Az első assemblerek nemigen voltak képesek többre, mint az utasítás- és regiszter-mnemonikok bináris kóddá történő átalakítására. Az azóta eltelt idő alatt azonban az assembler programok sokat fejlődtek a gyakorlati igényeknek megfelelően. Az első nyilvánvaló igények közé tartozott a szimbólumok használata konkrét címek, B/K eszközök és utasításszekvenciák használatakor. Ezek az úgynevezett szimbolikus assemblerek lehetővé teszik a programozó számára saját nevek/elnevezések (azaz szimbólumok) használatát a programban. Az assembler ezen kívül sok egyebet is tesz:

— Elvégzi az adatok és címek különböző számrendszerekből bináris számokká történő konvertálását, valamint a karaktereknek ASCII vagy EBCDIC (IBM) kódokká történő konvertálását.

— Lehetőséget biztosít egyszerű számítások elvégzésére a fordítás részeként.

— Információt nyújt a töltő számára a program részeinek és az adatoknak a memóriába történő elhelyezéséhez.

— Lehetővé teszi a programozó számára memóriaterületek ideiglenes adat-

> Nehogy azt higgye valaki, hogy az itt leírtak a ködbe vesző régmúltra vonatkoznak. A mikroprocesszoros alkalmazásokban még a nyolcvanas évek elején sem volt ritka (különösen beépített vezérlők, intelligens műszerek és mikroprocesszoros berendezések esetén) a hexadecimális programozás és tesztelés alkalmazása, mivel ezekben az esetekben nem volt lehetőség futtatórendszer és segédprogramok (assembler, debugger...) elhelyezésére a memóriában a memória és a szükséges fejlesztőrendszer költségei miatt. (Bizony, bizony: Nyugaton a programozókat is jól fizetik, ezért ott a programozói munka igen jelentős költséget jelent a büdzsében.)
> A memóriaáramköröknek a nyolcvanas évek közepén bekövetkezett árcsökkenése, a fordítóprogramok minőségi javulása és a speciális célú processzorok teljesítményének növekedése lehetővé tette olyan magas szintű nyelvek használatát a beépített vezérlők programozására, mint az ADA, a C, a Forth és a Modula-2, ami jelentősen megkönnyítette a fejlesztők dolgát, és lecsökkentette a fejlesztési költségeket (a magas szintű nyelveken történő fejlesztés ugyanis gyorsabb és ezért olcsóbb is, mint az Assembly használata).

tárolóként való használatát és fix adatok elhelyezését a memóriában.

— Lehetőségeket biztosít program- vagy eljáráskönyvtárak használatára.

— Lehetővé teszi a programlista formátumának meghatározását és az alkalmazott B/K egységek megválasztását.

Ma már az assemblerek szinte kivétel nélkül a szimbolikus assemblerek családjába tartoznak.

Példaprogramunk, ami egyébként a memóriában tárolt két számot adja össze és az eredményt egy harmadik rekeszben helyezi el, szimbolikus assemblerrel a következőképpen fest:

```
ORG 0200H
CIM1 EQU 0260
CIM2 EQU 0261
CIM3 EQU 0263
MOV AL, [CIM1]
ADD AL, [CIM2]
MOV [CIM3], AL
```

Ez azért már „szinte emberi fogyasztásra is alkalmas" kód.

Van egy púpú, van két púpú, sőt több...

Egy másik, az ismétlődő kódrészletek behelyettesíthetésére vonatkozó igény kielégítésére hozták létre az úgynevezett makrókat. A makrók segítségével több utasításnak — mondjuk egy logikai vagy műveleti egységnek — nevet adhatunk, és a programban később ezzel a névvel hivatkozhatunk rájuk. Az assembler a fordítás során a megfelelő utasításokat behelyettesíti a hivatkozási helyekre. A makrók a programok olvashatóságát megkönnyítik, a programozást gyorsabbá teszik. Manapság szinte minden assembler rendelkezik makróképességekkel (ún. macroassemblerek).

A makrók használata során merült fel az az igény, hogy a program bizonyos részei csak valamilyen feltételtől függően kerüljenek be a tárgykódba. Ezt a feltételes direktívák használata teszi lehetővé. A feltételek persze nem futásidejűek (tehát nem a program végrehajtása során keletkeznek), hanem fordításidejűek, azaz a fordítás kezdetekor, illetve fordítás közben adhatók meg. Bár a feltételes Assembly kiterjeszti a programozó lehetőségeit, használatával mégis csínján kell bánni, mivel túlzott használata a programot olvashatatlanná (vagy inkább követhetetlenné) teszi.

Az assemblereknek ezenkívül más csoportosításuk is van. Az ún. self-assembler vagy resident-assembler azon a gépen fut, amelyikre a programot készítjük. Ezzel ellentétben a cross-assemblerek másik gépen futnak, mint amelyikre a kódot generálják. Tipikus példa a cross-assemblerek használatára a beágyazott vezérlők programozása és fejlesztése, amikor a célgépre egy teljes fejlesztői rendszer elkészítése költséges lenne, ezért célszerűbb egy másik, már meglévő rendszeren végezni a fejlesztést. Például nagyszámítógépen vagy PC-n. A cross-assemblereket általában magas szintű nyelveken írják, így lehetőség van az ún. host gép (az a gép, amelyiken a fejlesztés folyik) rugalmas változtatására. (Az a rendszer, amelyikre a fejlesztés történik, az ún. target, azaz célrendszer.) A macroassembler nevet a makrókat kezelni tudó assemblerek viselik. A microassembler viszont olyan speciális assembler, amely lehetővé teszi a processzor utasításait definiáló mikrokód előállítását. A metaassembler képes több processzortípus számára tárgykódot generálni. Természetesen a célprocesszor típusát fordítás előtt meg kell adni. A metaassemblerek bizonyos típusai lehetővé teszik egy kvázi Assembly nyelv használatát, amit azután a célgép utasításaira optimálnak, s ez nagyban megkönnyíti a többféle processzortípusra történő fejlesztést. A one-pass assemblerek a forráskódot egy menetben képesek tárgykóddá alakítani, míg a two-pass assemblerek ezt két lépésben végzik (az első körben az assembler összegyűjti és definiálja az összes szimbólumot, a második körben azután a hivatkozási helyekre behelyettesíti az aktuális értékeket).

És ez mind-mind kevés...

Sajnos az Assembly és az assembler ismerete még csak a fele utat jelenti a konkrét Assembly programozás felé. A programozónak a processzor alapos ismeretén kívül jól kell ismernie a konkrét rendszert is, amire a programot írja, ami egyszerű esetben is áramkörök sokaságának, valamint a regiszternek és a tárkiosztásnak az ismeretét jelenti. Ha az illető feladata az operációs rendszer vagy monitor programozása, akkor az ismereteknek nagyon alaposaknak kell lenniük, a hardver működésével maximálisan tisztában kell lennie (például tárkiosztás, regisztercímek, B/K egységek, bővítőkártyák felépítése stb.). Ha a gépen már van működtető rendszer, akkor célszerű ismerni a rendszer működését és a rendszerhívásokat, amelyek segítségével a rendszer szolgáltatásai elérhetők (például PC-n a DOS- és BIOS-hívások ismerete).

A programozáshoz szükséges még az adatok ábrázolására vonatkozó szabályok ismerete (kettes komplemens bináris, ASCII karakter kódok, lebegőpontos aritmetika stb.). Ezen felül természetesen — mint minden esetben — a programozónak rendelkeznie kell a feladat megoldásához szükséges információk halmazával (elvárások, hardver-szoftver peremfeltételek...) és azt használni is kell tudnia. Jól látható, hogy az assembly-programozó feladata nem könnyű, de mindenesetre izgalmas. A magas szintű nyelvek a programozó feladatát megkönnyítik azáltal, hogy a fentiek egy részét a programozó elől elrejtik. De ne felejtsük: a legabsztraktabb magas szintű műveletet is gépi kódú utasításként végzi!

Külön kaszt

A cikk szerzőjének módjában állt az Assembly-programozásnak a fentiekben felsorolt minden esetét élesben kipróbálni. Elmondhatom, hogy az Assembly-programozásnak mind a mai napig megvan a varázsa és izgalma (a hibalehetőségekről nem is beszélve). Ezért nem minden ok nélkül alkotnak az assembly-programozók külön kasztot a programozók között. A varázs ellenére azonban az Assembly-programozás nem könnyű, és használatát nem tudom tiszta szívvel ajánlani a hobbiból programozóknak. Akiket még ezek után sem sikerült elriasztanom az Assembly-programozástól, azoknak ajánlom a i86 processzorok gépkönyveit, illetve a processzorok programozásáról magyarul is megjelent könyveket, az IBM PC-s gépkönyveket, a DOS-kézikönyveket és Peter Norton fantasztikusan jó könyveit az IBM PC-k programozásáról, amelyek közül kettőnek a magyar fordítása is kapható. Kedvcsinálóként (vagy riasztásul?) a lemezen található néhány bőven kommentált Assembly nyelvű program.

Villányi László

ALAPJÁRAT

rovatunk ez alkalommal teljes egészében átköltözött a mágneslemez mellékletre, ahol Kovács P. Attila GEM operációs rendszert ismertető sorozatának magyarázó szöveg nélküli, programlistás XIV. része megtalálható.

Saját célok szolgálatában

Könnyedén, rugalmasan, elegánsan

Eddig a Modula-2 elemi típusaival ismerkedhettünk meg: az egész, a természetes, a valós, a karakter és a logikai típusokkal. Mindegyik standard típusra egy standard típusazonosítóval hívatkozhatunk: INTEGER, CARDINAL, REAL, CHAR és BOOLEAN. A Modula-2 ezen felül lehetővé teszi a programozó számára új típusok definiálását is. Ezek közül ebben a részben a felsorolt és az intervallum típusokkal foglalkozunk.

Egy programozó által definiált típus bevezetése egy modulba (illetve a modul-univerzumba) általában két lépésből áll. Az első lépésben a típusdeklarációban a típus „tulajdonságát" írjuk le, és azonosítóval látjuk el (legyen neve a gyereknek). Ezután új típusú változókat ugyanúgy deklarálhatunk, mint elemi típusú változókat, azaz a típusazonosító segítségével. A változódeklaráció határozza meg a változó típusát és így az állandó tulajdonságait. A típus lehet valamelyik elemi típus, illetve egy a modulban deklarált vagy könyvtármodulból importált típus.

Strukturált vagy nem strukturált?

Mint láttuk, a típusdeklarációkat a TYPE kulcsszó vezeti be. A típusok két osztálya a strukturált és nem strukturált típusoké. Minden típus meghatározza azoknak az értékeknek a halmazát, amelyeket az adott típusúnak dekralált változók felvehetnek. A nem strukturált típusok értékei atomiak, míg a strukturált típusok komponensekből állnak. A CARDINAL típus például nem strukturált, így elemei atomiak. Nincs sok értelme ezek után 255 harmadik bitjére hivatkozni: az a körülmény, hogy egy számnak van harmadik bitje vagy harmadik számjegye, a típus belső reprezentálásának jellegzetessége, amelyet a Modula-2 szándékosan elrejt. Az eddig bemutatott elemi típusok a BITSET kivételével mind strukturálatlan típusok voltak. A felhasználó által definiált felsorolás és intervallum típusok szintén nem strukturált típusok, míg a BITSET és a később bemutatandó halmaz, tömb és rekord típusok strukturált típusok.

Általában ha egy T típust deklarálunk (TYPE T = egyTípus), és deklarálunk egy T típusú t változót (VAR t : t), akkor a két deklarációt mindig összevonhatjuk (VAR t : egyTípus). Ebben az esetben azonban t típusának nics explicit módon deklarált neve, ezért anonim marad. Mivel a típuskonfliktusok feloldásában (típusátalakítás) és ahogy azt később látni fogjuk, a konstansdeklarációkban és strukturált érdékadásokban a típus nevének ismerete szükséges, ezért célszerű az újonnan létrehozott típusokat nevesíteni.

A típuskoncepció nagyon fontos, mivel egy program változóinak halmazát diszjunkt osztályokra osztja. Az eltérő osztályokhoz tartozó elemek összerendelését ezért már a szintaktikai ellenőrzés során ki lehet szűrni. (A példákat a lemezmellékleten lévő M2-04.LST fájl 1. listája tartalmazza.)

A valós életet szimuláló alkalmazásokban az eddig megismert elemi típusokon kívül szinte végtelenül sok, a különböző problémák megoldásához szükséges típusra lenne szükség. Természetes, hogy nincs olyan programozási nyelv, amely ezeket mint előre definiált standard típusokat tartalmazná. Ugyanakkor egy magas szintű nyelvnek rendelkeznie kell olyan eszközökkel, amelyek segítségével a programozók saját céljaikra könnyen, rugalmasan és elegáns formában képesek létrehozni új típusokat. Ezeknek az eszközöknek a megfelelő kiválasztása kényes feladat, mivel ha kevés van belőlük, vagy a célnak nem megfelelőek, akkor nem támogatják hatékonyan a programozó munkáját, ha pedig túl sok, akkor előállhat a bőség zavara. A további típusok az új típusok definiálására szolgálnak.

A felsorolt típusok

Tételezzük fel, hogy programunkban személyek nemét és szemük színét kell kezelnünk (útlevél-kezelés). A nemet jelölő változót (nem) definiálhatnánk CHAR típusúnak (VAR nem: CHAR), mikor is „F" jelentené a férfit és „N" a nőt. Igen ám, de mit történik akkor, ha a nem változóhoz valamilyen programhiba miatt mondjuk az „a" betű rendelődik hozzá? Mivel változónk CHAR típusú, 0C-től 377C-ig tetszőleges értéket vehet fel. Jobb megoldást jelent az eleve kételemű BOOLEAN típus használata. Ebben az esetben célszerű változónkat mondjuk ferfi-nek elnevezni, így a TRUE érték jelölné a férfiakat és a FALSE a nőket.

Nézzük most a szemek színét. A szemeSzine nevű változót deklarálhatjuk INTEGER típusúnak úgy, hogy a lehetséges színeket (fekete, barna, kék, zöld) egy egy értékkel jelöljük (0, 1, 2, 3). Ez a megoldás nem szerencsés, mivel a programkódban az IF szemeSzine = 2 THEN ... utasítás nem teszi nyilvánvalóvá, hogy a kék színre gondoltunk. Megoldást állandók deklarálása jelenthetne: CONST Fekete=0;Barna=1;Kek=2;Zold=3.

Ezek után az utasítás már beszédesebb: IF szemeSzine=Kek THEN ... A probléma most az, hogy a szemeSzine változónkat CARDINAL típusúnak kellene deklarálnunk, és így minden pozitív egész számra értelmezett műveletet elvégezhetünk vele, például a szemeSzine:= szemeSzine*100 kifejezés megengedett, de nincs értelme, mivel a

kapott eredmény kimutat a definiált színek halmazából.

Az ilyen jellegű programozási feladatok és problémák megoldására a Modula-2 felsorolt típusait célszerű használni. Egy felsorolt típust (ami egyébként nem strukturált típus) úgy definiálunk, hogy felsoroljuk a típushoz tartozó értékek halmazát (innen az elnevezés). A T = (a1,a2,a3,...,aN) típus deklarációval bevezetjük T-t mint új nem strukturált típust, amelynek lehetséges értékeit N darab konstansazonosító (a1,..aN) jelöli. Kizárólag ezek az értékek tartoznak ehhez a típushoz. A felsorolt típus formális definíciója:
$ FelsoroltTípus = „(" AzonosítóLista„)".

A példában szereplő feladatot most már két új felsorolt típus segítségével oldhatjuk meg:

TYPE Nem=(Ferfi, No); SzemSzin=(Fekete,Barna,Kek,Zold);
illetve
VAR neme:Nem; szemeSzine:SzemSzin;
és a programban:
neme:=Ferfi;...szemeSzine:=Zold;..IF neme=No THEN...

TYPE HaziAllat=?

A felsorolt típusok használatakor a következőkre kell figyelni:

1. Egy felsorolt típus értékei nem karakterláncok, ezért nem írhatók ki és nem olvashatók be. A fenti példával WriteString(Zold) nincs megengedve.

2. Egy adott felsorolt típusú változó és a felsorolt típus értékei között az összehasonlítás meg van engedve. Például az IF szemeSzine=Kek THEN... kifejezés megengedhető.

3. Összehasonlítás és értékadás két eltérő felsorolt típusú változó, illetve érték között tiltva van. Így a következő két utasítás — szerencsére — nincs megengedve:
neme := Fekete; IF szemeSzine=No THEN...

4. Felsorolt típus nem szerepelhet művelet operandusaként. A szemeSzine := Kek — Barna; kifejezés ezért nincs megengedve. Ez azt jelenti, hogy a felsorolt típusokra értelmezett műveleteket a programozónak kell definiálnia. Az egyedüli értelmezett műveletek az értékadás, illetve az összehasonlítás (lásd később).

5. A felsorolt típus értékei konstansok, és ezért természetesen nem rendelhetünk hozzájuk értéket. A Ferfi:=No ugyanúgy hibás, mint a 3:=4; kifejezés.

6. Ugyanaz az érték nem szerepelhet egy modulon belül két különböző felsorolt típusban. Például a
TYPE Fa=(tolgy,bukk,meggy,szilfa);
Gyumolcs=(eper,malna,meggy);
típusdeklaráció hibás, mivel meggy mind a Fa, mind a Gyumolcs deklarációjában szerepel. Ez egyben azt is jelenti, hogy a felsorolás típusokban szereplő azonosítók az adott blokkon belül más célra (konstans, típusváltozó, eljárás neve) nem használhatók fel.

Ahogy azt a 4. pontban láttuk, a felsorolt típusok összehasonlíthatók. Ennek az az oka, hogy a felsorolt típusok rendezettek (sorszámozottak). A legkisebb az elsőnek deklarált azonosító (a1), a legnagyobb a legutolsónak deklarált (aN). A felsorolt típusban elsőnek deklarált konstans azonosító sorszáma mindig 0, a többi azonosító sorszáma a listában elfoglalt helyétől függ. Például:
TYPE HaziAllat=(Tyuk,Kacsa,Liba,Sertes,Lo) esetén az elemek közötti reláció Tyuk<Kacsa<Liba<Sertes<Lo, illetve az elemek sorszáma Tyuk 0,Kacsa 1,Liba 2,Sertes 3,Lo 4. Ez a rendezési reláció lehetővé teszi a felsorolt típus értékeinek összehasonlítását. Például:

```
VAR kedvenc : HaziAllat;...
  IF kedvenc> Liba THEN
    WriteString ( „sertés, ló" )
  END
```

Mivel a felsorolt típusok értékei rendezettek, ezért a már korábban megismert ORD függvény itt is használható, akárcsak az INC és DEC eljárások. Például:
ORD(Kacsa) = 0, ORD(Lo) = 4,
INC(Kacsa) = Liba, DEC(Lo,3) = Kacsa (Micsoda genetikai szenzáció!!).

A karakter típusnál láttuk, hogy a CHR függvény az ORD inverze. CHR(x) az a karakter, amelynek sorszáma x. A CHR függvény nincs definiálva felsorolt típusokra, azonban a VAL(T,X) standard függvény segítségével bármilyen sorszámozott típust előállíthatunk sorszáma (X) és típusa (T) segítségével. A fenti deklarációkat felhasználva:
VAL(HaziAllat,3) = Sertes
VAL(HaziAllat,0) = Tyuk
VAL(SzemSzin,1) = Barna
VAL(Nem,1) = No

A VAL természetesen értelmezhető CHAR, INTEGER és CARDINAL típusokra is, azonban az alábbi összefüggések miatt nem valami hasznos:
VAL(CARDINAL,n) = n
VAL(CHAR,n) = CHR(n)
VAL(INTEGER,n) = n ha n>=0

Az elemi típusok közül a BOOLEAN speciális előre definiált felsorolt típusnak tekinthető az alábbi definícióval:
TYPE BOOLEAN=(FALSE,TRUE)

Ez a fentiek figyelembevételével a következőket jelenti:
1. FALSE <TRUE
2. ORD(FALSE) = 0,
ORD(TRUE) = 1
3. VAL(BOOLEAN,0) = FALSE,
VAL(BOOLEAN,1) = TRUE
4. Ha X FALSE, akkor értéke TRUE lesz INC(X) végrehajtása után.

Ha X TRUE, akkor értéke FALSE lesz DEC(X) végrehajtása után.

Ha X TRUE, akkor INC(X) hibát okoz.

A Pascal felsorolt típusa megegyezik a Modula-2-évei, a C nyelvben eredetileg (K&R C-ben) nem szerepelt felsorolt típus, később azonban a C kulcsszavakat kibővítették az enummal felsorolt típusok létrehozására. Az enum már az ANSI C része. Mindezek ellenére a C enum típusa nem annyira markáns, mint a Pascal-szerű nyelveké a C kevésbé szigorú típuskonvenciója miatt, valamint azért, mert a C enum típusa nem feltétlenül tekinthető folytonos sorszámozott típusnak. Például:
enum szivarvany{voros,narancs,sarga=10,zold,kek=0,lila,ultra=10}, itt a sorszámok: 0,1,10,11,0,1,10 !!! Az ilyen és hasonló deklarációk nemigen nevezhetőek felsorolásnak. Ember legyen a talpán, aki ebből a zavaros felsorolásból a programkódban képes az összefüggéseket kihámozni, főleg ha csak annyit tud, hogy az értékek úgynevezett „felsorolt" típushoz tartoznak. (Példák a lemezen: az M2-04.LST fájl 2. listája és EX04-01.MOD, EX04-02.MOD programok.)

Az intervallum típusok

Az egész, pozitív egész, karakter, felsorolás és intervallum típusokat sorszámozott típusoknak nevezzük. A felsorolás és intervallum típusok programozó által létrehozott sorszámozott típusok. A JPI Modula-2 terminológiájában megkülönböztetjük a rövid sorszámozott típusokat, amelyek nem tartalmazzák a LONGCARD és LONGINT típusokat, ennek a megkülönböztetésnek a halmazok definíciójánál lesz jelentősége.

Egy már definiált sorszámozott típus (elemi vagy programozó által definiált) tetszőleges részsorozatát deklarálhatjuk külön típusként. Az ilyen, programozó által definiált típust a deklarálás módja alapján intervallum típusnak nevezzük,

mivel a típusdeklaráláskor két szélsőértéket adunk meg. Az intervallum típusok egy sorszámozott típus folytonos részhalmazát jelölik ki, ez a sorszámozott típus az intervallum típus úgynevezett alaptípusa. Az intervallum típusok alaptípusai a típus-ellenőrzési és típuskonverziós szabályok miatt fontosak. Az alaptípusra definiált minden művelet alkalmazható az intervallum típusra is. Egyetlen kivételként az intervallum típusú változókhoz hozzárendelhető értékeket kell számon tartani. Ennek egyik következménye az, hogy egyes Modula-2-implementációk nem engedik meg intervallum típusú változók használatát az InOut modul beolvasó eljárásaiban. A Modula-2-fordítók nagy része azonban nem ilyen szigorú, viszont ha a beolvasáskor — terminálról vagy fájlból — az aktuális érték a definiált intervallumon kívül esik, futásidejű hiba keletkezik. Egyébként az intervallum típusú változók által felvehető értékek szempontjából a definiált intervallum zártnak tekintendő, azaz a határértékeket is felveheti a változó.

Formálisan:$ IntervallumTípus=[kvalident] "["ÁllandóKifejezés".."ÁllandóKifejezés"]".

Érdemes megfigyelni, hogy a határértékeket állandó kifejezéssel is megadhatjuk, azaz a Pascaltól eltérően egy állandó helyett egy teljes komplex kifejezés is állhat, feltéve, hogy az értéke állandó. Ez az eltérés a Modula-2 és a Pascal között minden olyan esetben igaz, ahol a Pascalban állandó használható. Természetesen a határértékeket definiáló kifejezések típusának azonosnak kell lennie. Ugyanilyen nyilvánvaló követelmény az is, hogy az alsó határértéknek definált kifejezés sorszáma kisebb legyen, mint a felső határértéknek definiált kifejezésé.

Példák

TYPE Szarnyas=[Tyuk..Liba] (Az alaptípus felsorolt típus, ilyen esetekben az alaptípust természetesen korábban már definiálni kellett, úgy, hogy Tyuk<Liba összefüggés igaz legyen.)

KisEgesz=[-10..10] (Az alaptípus INTEGER.)
KisTermeszetes=[0..10] (Az alaptípus CARDINAL.)
Szamjegy=['0'..'9'] (Az alaptípus CHAR.)

A KisValos=[0.0..10.0] deklaráció hibás, mivel a REAL nem sorszámozott típus, hiszen a valós számok számossága végtelen még a gépfüggő határértékeken belül is, igaz ugyan, hogy a számítógépekben csak véges felbontásban ábrázolhatóak a valós számok, de az absztrakció ezt nem veheti figyelembe.

Az alaptípus megállapítása egyedül pozitív egész számok esetén nem egyértelmű. A KisSzam=[0..10] definíció alapján a fordítók alaptípusnak a CARDINAL-t tekintik, mivel ha alapértelmezésben az alsó határérték pozitív vagy nulla, az alaptípus CARDINAL, ha pedig negatív, akkor INTEGER. Amennyiben ezt felül akarjuk bírálni, lehetőség van az opcionális kvalifikált azonosító segítségével az alaptípust definiálni (KisSzam=INTEGER[0..10]). Ennek az opcionális alaptípus-definiálásnak a jelentőségét bővebben az M2-04.LST fájl 3.listája illusztrálja. Példák a lemezen: az M2-04.LST fájl 4. listája és az EX04-01.MOD, EX04-02.MOD programok.

Az intervallum típus használatának előnye az, hogy további ellenőrzésre ad lehetőséget nem megfelelő értékek hozzárendelésekor, és így nagyban elősegíti az esetleges hibák detektálását. Ezért célszerű — amikor csak lehet — intervallum típusokat alkalmazni. Az ilyen esetekben egyébként a futásidejű hibák a jellemzőek, azaz azok az esetek, amikor a felhasználó ad meg a határértékeken kívül eső paramétereket, vagy mikor adatfeldolgozás során az adatfájlokban találhatók hibás adatok. Persze az, hogy a program „run time error..." hibával áll le, még nem vigasztalja a végfelhasználót. Az ilyen jellegű hibafelderítés jelentősége abban keresendő, hogy ezek segítségével készíthetünk olyan programokat, amelyek golyóállóak és bolondbiztosak. Ezért a tesztelési fázisban célszerű a maximális hibaellenőrzési opcióval generálni a programokat, és az esetlegesen felmerülő hibákat a programban kiküszöbölni. Az ilyen módon tesztelt és javított programot aztán már „slankíthatjuk" a gyorsaságra optimalizálva. Amennyiben nem követünk el mindent, hogy a hibákat idejében felfedezzük, akkor a programunk nagyon hosszú ideig működhet látszólag hibátlanul, és amikor a mulasztások kiderülnek, egyrészt a hiba valódi okának megállapítása lesz nagyon nehéz, másrészt esetleg értékes adatok sokaságát tehetjük tönkre rosszul tesztelt programunkkal, aminek jogi és anyagi következményeit is nekünk kell vállalni (bár ez Magyarországon még nem jellemző), nem beszélve saját presztizsünkről.

Villányi László

Kocsis Zoltán — Gazsó Zoltán: PC-START (Az IBM-kompatibilis személyi számítógép kezelése kezdőknek)

(Budapest, 1991, Innotech Műegyetemi Innovációs Park Kft., 99 oldal, 195 Ft)

Sokan vannak, akik eddig úgy tekintettek a számítógépre, mint egy titokzatos, intelligens szerkezetre, és most munkájukhoz kapcsolódóan vagy egyszerűen csak kíváncsiságból szeretnék közelebbről megismerni. Ehhez nyújt segítséget a PC-START című könyv, elsősorban azok számára, akik még soha nem kerültek közvetlen kapcsolatba a számítógéppel, és lehetőleg sikerélményeken keresztül szeretnének megismerkedni vele. Azoknak is jó segítőtársa lesz, akik már megtették az első lépéseket ebben az irányban, de rendszerezett ismereteket akarnak szerezni.

A könyv a legelterjedtebb IBM-kompatibilis számítógépek alapfokú kezeléséhez szükséges ismeretanyagot tartalmazza. Megtudhatjuk, mi a jelentése néhány olyan fontos számítástechnikai kifejezésnek, mint a szoftver, hardver, bájt stb., hogyan épül fel és milyen misztikus parancsoknak engedelmeskedik az eddig misztikus doboznak hitt számítógép. A kezdő felhasználók számára legfontosabb parancsok leírása részletes, használatukat számos példa mutatja be. A számítógép általában angolul kommunikál, ezért a gyakran előforduló üzenetek magyar fordítása megtalálható a függelékben.

A könyv jól alkalmazható a számítástechnika oktatásában segédanyagként, hiszen a szerzők azokra a tapasztalatokra építettek, amelyeket a kezdő felhasználók oktatásában az elmúlt 4 évben a Budapesti Műszaki Egyetemen szereztek.

Új könyvesbolt a Belvárosban

Lapunk 1990. szeptemberi számában *Olvasni jó! — volna...* című írásomban vázoltam a hazai szakirodalmi kínálat nem éppen szívderítő helyzetét. Bár a magyar nyelvű szakkönyvek kiadása körül nem sok fejlődés tapasztalható, az idegen nyelvű szakirodalom beszerzési lehetőségei jelentős mértékben javultak. Egyre több csatornán keresztül lehet külföldi szakkönyvekhez és folyóiratokhoz hozzájutni.

1991. június 20-án az *Akadémiai Kiadó* új, idegen nyelvű könyvesboltot nyitott a Gerlóczy utcában, a *Famulus* Könyvesbolt helyén, *Academic Bookshop* néven. Itt a kiadó idegen nyelvű kiadványain kívül nagy nyugati kiadóknál megjelent könyveket is találhatunk, a tudomány minden ágából. Bőséges a választék a *számítástechnikai könyvekből* is. Megtaláljuk a magyar szakemberek által is ismert *McGraw-Hill* vaskos köteteit, a *Carl Hanser Verlag* sorozatait, a *Macmillan*, a wiesbadeni *Vieweg* és más kiadók számítástechnikai és rokon témájú kiadványait. Az — ugyancsak világhírű — angol *Oxford University Press* és a *Penguin* szótárakkal (számítástechnikai szakszótárakkal is) képviselteti magát.

A külföldről behozott szakkönyvek árai viszonylag mérsékeltek. A forintár a külföldi katalógusárral egyezik meg, mindig a könyvnek a boltba érkezésekor érvényes hivatalos árfolyamot alapul véve. Elsősorban könvtárak, oktatási intézmények rendeléseire számítanak. A bolt készletét folyamatosan felfrissítik, ha egy megrendelt könyv nem áll rendelkezésre, beszerzik. Az igények gyors, pontos kielégítését a számítógépes nyilvántartás is segíti.

(ba-ló)

Bibliográfia

Összeállításunk első fele az *Academic Bookshop* — július harmadik hetében elérhető — számítástechnikai kínálatából ad ízelítőt. (Az idő előrehaladtával az itt felsorolt választékban és az árakban eltérések lehetnek.) Az ajánló bibliográfia második fele A hónap témájához kínál szakirodalmi segédletet.

Angol nyelvű könyvek

Baritz, Tony — Dunne, David: AS/400 (Concepts and Facilites)
McGraw-Hill, 1990. 279 oldal. Ára: 4681 Ft.

Campbell, Mary: Harvard Graphics 2.3 (Made Easy)
Osborne/McGraw-Hill, 1990. 548 oldal. Ára (lemezmelléklettel): 2370 Ft.

Chandor, Anthony: The Penguin Dictionary of Computers
Penguin, 1985. 488 oldal. Ára: 383 Ft.

Dean, David: Using Microsoft Word for Windows
Osborne/McGraw-Hill, 1990. 594 oldal. Ára: 2300 Ft.

Feibel, Werner: Using ANSI C in UNIX
Osborne/McGraw-Hill, 1990. 626 oldal. Ára: 3196 Ft.

Hordeski, Michael F.: The Illustrated Dictionary of Microcomputers
TAB BOOKS, 1990. 442 oldal. Ára: 2434 Ft.

i486 Microprocessor (Programmers Reference Manual)
Intel — Osborne/McGraw-Hill, 1990. Ára: 2940 Ft.

Luther, Arch C.: Digital Video in the PC Environment
Intertext Publications — McGraw-Hill, 1991. 346 oldal. Ára: 2940 Ft.

McElreath, T. Jack: IMS (Design & Implementation Techniques)
McGraw-Hill Book Company, 1990. 238 oldal. Ára: 3837 Ft.

O'Brien, Stephen K.: Turbo Pascal 6: The Complete Reference
Borland — Osborne/McGraw-Hill, 1991. 690 oldal. Ára: 2940 Ft.

Zuse, Horst: Software Complexity (Measures and Methods)
Walter de Gruyter, 1991. 605 oldal. Ára: 6884 Ft.

Német nyelvű könyvek

Hambeck, Klaus: Einführung in das Programmieren in Cobol
Walter de Gruyter, 1981. 163 oldal. Ára: 1220 Ft.

Kief, Hans B.: NC/CNC Handbuch '90
NC-Handbuch-Verlag, 1990. 560 oldal. Ára: 959 Ft.

Kopp, Herbert: Graphische Datenverarbeitung
Hanser Verlag, 1989. 218 oldal. Ára: 1656 Ft.

Marschner, Freimut: Quantenchemie mit Personal Computern (Eine Einführung mit interaktiven Programmen für IBM-PC und Kompatible)
Walter de Gruyter, 1991. 409 oldal. Ára (2 lemezmelléklettel): 4182 Ft.

Martin, James: Einführung in die Datenbanktechnik
Hanser Verlag, 1990. 384 oldal. Ára: 2527 Ft.

von Reeken, Jeil: Objektorientieres Programmieren mit C++
Hanser Verlag, 1991. 216 oldal. Ára: 2091 Ft.

Vollenweider, Peter: PostScript (Konzeption und Anwendungen)
Hanser Verlag, 1991. 212 oldal. Ára: 1917 Ft.

Copyright *.*

Ez a válogatás elsősorban a mélyebb történeti áttekintésre teremt lehetőséget az érdeklődőknek.

Magyarország Magánjogi Törvénykönyve (A m. kir. Igazságügyminiszter által 1928. március 1-jén az Országgyűlés elé terjesztett törvényjavaslat). M. kir. Igazságügyminisztérium, Budapest, 1928.

Szladits Károly: A magyar magánjog vázlata. Grill Könyvkiadó, Budapest, 1933.

Marton Géza: A római magánjog elemeinek tankönyve. Institúciók
Tankönyvkiadó, Budapest, 1963.

Benedek, K. — Világhy, M.: A Polgári Törvénykönyv a gyakorlatban.
Közgazdasági és Jogi Könyvkiadó, Budapest, 1965.

A szerzői jog kézikönyve (Szerk.: Benárd Aurél és Tímár István)
Közgadasági és Jogi Könyvkiadó, Budapest, 1973.

Boytha, Gy.: A számítógépi programok alkotásához fűződő érdekek jogi védelme. Gazdaság és Társadalom (Az MTA IX. Osztály Közleményei) Budapest, 1977.

Jacsó Péter: A szoftver jogi oltalma. KSH-SZÁMOK, Budapest, 1981.

Thorne, D.H.: The legal guide to computer software protection. A Practical Handbook on Copyrights, Trademarks, Publishing and Trade Secrets. Prentice-Hall Inc., New Jersey, 1985.

Gyertyánfy, P. — Perjés, S.: A szoftver szerzői jogvédelméről — magyarul. KERSZI, Budapest, 1986.

A software szerzői jogvédelmének kérdései (A MIE 1986. május 19-21. között rendezett software tanfolyamán elhangzott előadások szövege). Magyar Iparjogvédelmi Egyesület, Budapest, 1987.

Gyertyánfy Péter: A számítógépi programok és elektronikus adattárak szerzői joga (Kandidátusi értekezés). Budapest, 1989.

REX TRADE

Kereskedelmi és Szolgáltató Kft.

Másológépek

Kellékek

Festékek

Alkatrészek

A/3, A/4, A/5 papírok

Pauszmásológépek

Telefaxok

Cím: REX TRADE Kereskedelmi és Szolgáltató Kft.
1139 Budapest Fáy u. 6.
Telefon: 1203-280/ 149-es, 156-os mellék, 1202-805

INFORMÁCIÓKÉRÉS: 25 ▼

ÚJ! Az új, bővített ÚJ!

CLIPPER 5.01

kapható más, érdekes
Clipper kiegészítőkkel
(dGE, Netlib, dbdBAK, Tools II. stb.)

Tegye Clipper 5.0-ját valódi
objektumorientált programozási nyelvvé
a Chydale OOPS Extension-nel!
Magyarországon csak nálunk kapható.

Kérjen részletes információt!

R-SOFT-SZENZOR

Budapest
Telefon: 201-6891, Fax: 201-8619
1277 Budapest 23, Pf. 45.

INFORMÁCIÓKÉRÉS: 13 ▼

SZÁMÍTÁSTECHNIKA KULCSRAKÉSZEN!

- XT, AT, 386, 386SX, 486, Laptop minden kiépítésben.
- EPSON, STAR, NEC nyomtatók teljes választéka.
- MODEMEK és egyéb tartozékok széles választéka.
- Magánszemélyeknek KÉSZPÉNZFIZETÉS ESETÉN KEDVEZMÉNY!
- ASHTON-TATE, BORLAND, MICROSOFT, NANTUCKET, LOTUS szoftverek
- SHAREWARE-programok (1200-féle) 360,– Ft + ÁFA áron.
- MODEMES távadat-átviteli és BBS-rendszerek szállítása.
- VÍRUSÖLŐ program (120-féle vírust öl!)
- NOVELL HÁLÓZATI SZOFTVEREK, hálózatkiépítés

Ajánlatunk:
NOTEBOOK: 386SX, 20MHz, 1,44 MB FDD,
LCD VGA (/40x480/16 szürke),
AKKUMULÁTOR, 3 kg 189.900,–Ft + ÁFA
AT számítógép: 1 MB RAM, 40 MB HDD,
1.2 MB FDD, Mono 14" (PHILIPS)
1 S, 1 P, 101 gombos bill. 64.900,– Ft + ÁFA
(Készpénzért 61.600,– Ft + ÁFA.)

Amikor ezt a hirdetést Ön olvassa, áraink már úgyis alacsonyabbak! Ezért kérjük, telefonáljon vagy írjon, és mi örömmel adunk felvilágosítást, küldünk részletes árjegyzéket!

QWERTY

High Tech. Kft.
1117 Budapest XI., Orlay u. 4.
Telefon: 166-3098, 185-2687, Fax: 185-2687
BBS: 118-7950 BUDAPEST BBS

NE FELEDJE: Nevünk ott található az Ön számítógépének billentyűzetén is!

INFORMÁCIÓKÉRÉS: 22 ▼

COMFORT

Szolgáltató, Kereskedelmi és Fejlesztő Kft.
holland–magyar vegyestársaság

1125 Budapest, Városkúti út 23/b.
Postacím: 1501 Budapest, Pf. 4.
Telefon/Telefax: (361)-173-3811
Fióktelep/iroda: 2040 Budaörs, Szabadság út 121.

Vége a kőkorszaki adatátviteli módszereknek, nem kell többé floppykat, mágnesszalagokat küldözgetnie két számítógép között.
ITT A

BLAST®

a világ kiemelkedően leghatékonyabb, legsokoldalúbb, 100%-osan hibamentes helyi és táv-adatátviteli szoftverterméke!
Kifejlesztette a Communications Research Group (USA). Hazánkban kizárólagos joggal terjeszti a **COMFORT Kft.**
A különböző verziók **már 21 000,– Ft-tól** kaphatóak! Szívesen adunk bővebb információt, kérjük, telefonáljon!

A BLAST kommunikációs szoftver jellemzői:

- 150-nél többféle számítógép/operációs rendszeren használható,
- 30-nál több modemtípust támogat,
- világszerte 50 000-nél több referenciahely,
- olcsó hardverigény a vonali összeköttetéshez,
- tömörített adatátvitel, automatikus hibakorrekció,
- teljes duplex módú vonalhasználat,
- különböző terminálemulációk,
- KERMIT, XMODEM, YMODEM és egyéb protokollok támogatása,
- rutin adatátvitelek automatizálhatósága,
- forgalmi naplózás,
- kezelése egyszerű, de magyar szakirodalom és tanfolyam is segíti a betanulást.

Ha még nem rendelkeznek a szükséges hardvereszközökkel, ezek beszerzésében is rendelkezésükre állunk.

Komplex számítógépes adatátviteli végpontok árai:

1. IBM PC XT, 640 kB RAM, 360 kB FDD, 20 MB HDD, mono monitor, 101 gombos billentyűzet, 2400 baudos modem, soros/párhuzamos port, BLAST kommunikációs szoftver ára: **69 900,–Ft**
2. MicroVAX-II, 5 MB RAM, 332 MB winchester, 80 MB szalagkazetta, 8 soros vonal, 2400 baudos modem, 5 VT-320-as terminál, 1–8 felhasználós VMS, BLAST szoftver ára: **2 699 900,– Ft**

A **BLAST** a Communications Research Group védjegye.
A **MicroVAX**, **VMS** a Digital Equipment Corporation védjegyei.
Az **IBM PC** az IBM védjegye.

AB DICK AEG OLYMPIA AGFA BROTHER FUJI GESTETNER KIS KODAK KONICA

BUSIMATIC CANON COPYER DEVELOP UBIX LANIER MINOLTA MITA NASHUA

FUJI GESTETNER KIS KODA... PANASONIC RANK XEROX

LANIER MINOLT... SHARP

OLIVETTI PANAS... INFOTEC

REX ROTARY SA... BROTHER

TRIUMPH ADLER INFOTEC SAVIN ... CANON COPYER DEVELOP

Molland® – PAPÍRREX

KFT.

1133 Budapest, Fáy u. 6.
Tel.: 120-2805,
120-3280/149 m.

AB DICK AEG OLYMPIA AGFA BROTHER FUJI GESTETNER KIS KODAK KONICA

BUSIMATIC CANON COPYER DEVELOP LANIER MINOLTA MITA NASHUA

ARJ: rögtön az élen!

Az Alaplap júliusi számában közölt „Tömörítők tesztelve" anyag még a tömörítés új csillaga, az ARJ nélkül elkészített teszt eredményét közölte. Időközben — most már az ARJ bevonásával — ismét próbának vetettük alá a tömörítőket, s az a „trónkövetelő" fényes diadalát hozta. Az Alaplap Könyvek sorozatban a tömörítésről hamarosan megjelenő kézikönyv már az új nyúzópróbák eredményét tartalmazza, de lényeges változásról lévén szó, érdemesnek tartjuk itt is közölni a teszt-összesítéseket.

1. *.DOC állományok (43 db, 429 236 bájt) tömörítése:

Program (+ paraméter)	Idő (sec)	Tömörített (bájt)	méret (%)	Lemaradás (%)
ARJ a	90	176 647	41,15	—
LHA a	113	178 964	41,69	1,31
PAK a	94	188 840	43,99	6,90
PKZIP –aex	60	191 232	44,55	8,27
PKZIP –aei	61	191 390	44,59	8,35
PKPAK –a	37	226 061	52,67	27,97
PKZIP –aes	35	226 346	52,73	28,13
ZOO –a	81	231 387	53,91	30,99
ARC a	64	244 452	56,95	38,38

2. *.EXE és *.COM állományok (21 db, 735 700 bájt) tömörítése:

Program (+ paraméter)	Idő (sec)	Tömörített (bájt)	méret (%)	Lemaradás (%)
ARJ a	151	465 670	63,30	—
LHA a	194	480 291	65,28	3,14
PKZIP –aex	106	502 155	68,26	7,83
PKZIP –aei	107	502 155	68,26	7,83
PAK a	162	504 386	68,56	8,31
PKPAK –a	75	605 344	82,28	29,99
PKZIP –aes	71	609 677	82,70	30,92
ARC a	127	621 501	84,48	33,46
ZOO –a	148	628 298	85,40	34,92

3. *.DBF állományok (5 db, 218 798 bájt) tömörítése:

Program (+ paraméter)	Idő (sec)	Tömörített (bájt)	méret (%)	Lemaradás (%)
ARJ a	30	14 229	6,53	—
LHA a	39	14 802	6,77	3,57
PKZIP –aex	78	17 208	7,86	20,40
PKZIP –aei	78	17 208	7,86	20,40
ARC a	15	20 355	9,30	42,42
PAK a	27	20 998	9,60	46,92
PKPAK –a	10	21 206	9,69	48,38
ZOO –a	18	21 655	9,90	51,52
PKZIP –aes	11	21 699	9,92	51,83

4. *.WK? állomány (1 db, 74118 bájt) tömörítése:

Program (+ paraméter)	Idő (sec)	Tömörített (bájt)	méret (%)	Lemaradás (%)
ARJ a	11,6	48 403	65,30	—
LHA a	10,9	50 337	67,91	4,00
PAK a	9,0	51 372	69,31	6,13
PKZIP –aex	5,6	54 208	73,13	11,99
PKZIP –aei	5,6	54 208	73,13	11,99
ZOO –a	7,4	69 317	93,52	43,21
PKPAK –a	6,7	69 513	93,79	43,61
PKZIP –aes	6,6	70 417	95,01	45,48
ARC a	7,5	71 116	95,95	46,92

5. Grafikus állományok (EPS, PCX, TIFF, IMG) tömörítése (7 db, 2 959 535 bájt):

Program (+ paraméter)	Idő (sec)	Tömörített (bájt)	méret (%)	Lemaradás (%)
ARJ a	1110	1 240 864	42	—
LHA a	630	1 251 690	42	0,87
PAK a	725	1 302 225	44	4,95
PKPAK –a	207	1 331 652	45	7,32
PKZIP –aes	213	1 346 188	46	8,49
ZOO –a	332	1 370 265	47	10,43
PKZIP –aex	784	1 375 567	47	10,86
ARC a	326	1 454 546	50	17,22

6. Tömörített állományok továbbtömörítése (1-1 ARC, PAK, ZIP, ZOO, LZH, ARJ állomány, 209 083 bájt):

Program (+ paraméter)	Tömörített (bájt)	méret (%)
LHA a	202 801	97,00
ARJ a	202 802	97,00
PAK a	207 375	99,18
PKZIP	209 083	100,00
PKPAK –a	209 277	100,01
ARC a	209 375	100,14
ZOO –a	218 612	104,56

7. Vegyes állományok tömörítése (23 file, 748 671 bájt):

Program (+ paraméter)	Idő (sec)	Tömörített (bájt)	méret (%)	Lemaradás (%)
ARJ a /m1 /jm /jh65000	196	475 036	63,45	—
ARJ a	168	476 994	63,71	0,41
LHA a	150	481 310	64,29	1,32
PAK a	144	497 882	66,50	4,81
PKZIP –aex	90	497 074	66,39	4,64
PKZIP –aes	69	564 958	75,46	18,93
PKPAK –a	80	565 538	75,54	19,05
ARC a	138	578 448	77,26	21,77
ZOO –a	168	595 411	79,53	25,34

A teszt eredményéhez néhány megjegyzés:

— A PKZIP-hez hasonlóan az ARJ is rendelkezik olyan lehetőségekkel, hogy a felhasználó írja elő, milyen tömörítési technikát alkalmazzon a program.

— A Huffman puffer méretének beállítása nemcsak gyorsít az ARJ programon, hanem az ARJ állomány még tömörebb is lesz pár százalékkal. Igy az LHA-t is megelőzi.

— Már megjelent az ARJ 2.xx verziója is, ami még tömörebb archívokat eredményez, ráadásul tovább gyorsult.

A végeredmény alapján a leginkább ajánlott programok a következők:

1. ARJ 2.10
2. LHA 2.12
3. PAK 2.51
4. PKZIP 1.10

A ZOO és az ARC program használata csak akkor ajánlható, ha azok különleges szolgáltatásaira van szükség. Mind tömörítésben, mind sebességben, mind az általános szolgáltatásokban alaposan lemaradnak. A PKARC és PKPAK pedig már elavult, érdemes helyettük áttérni valamelyik hatékony tömörítőre.

Nagy Gábor

„Egyéb javaslatai, észrevételei: ...."

Bár e sorok bebillentyűzésekor a júliusi számunkhoz mellékelt közvéleménykutató kérdőívnek az ajándéksorsoláson való részvételhez megszabott, augusztus 31-i beküldési határideje még messze van, a lapzártáig beérkezett mintegy 800 kérdőív tanulmányozása alapján máris „bedobunk" néhány olyan témát, amelyet olvasóink vetettek fel a kérdőív utolsó pontjában kifejtett véleményükkel.

Az egyértelműen dicsérő sorok ismertetésétől eltekintünk, bár be kell vallanunk, hogy jólestek. Távol áll azonban tőlünk az elbizakodottság, s ha voltak is, akik sommásan úgy foglalták össze véleményüket, hogy: „A lap úgy jó, ahogy van", mi sokkal kritikusabban nézzük munkánkat és állandóan keressük a jobbítás lehetőségeit.

Az olvasói vélemények még alapvető kérdésekben is sokszor ellentmondóak. Vannak például, akik szószátyárnak tartják az Alaplapot, mások meg éppen a szakbarbárságtól mentes, oldottabb stílust hozzák fel egyik legfőbb erényünknek. Egyik olvasónk a hangvételre és a fel-felbukkanó karikatúrákra célozva felteszi a kérdést: „Vicclap vagy szaklap akar-e lenni az Alaplap?". Furcsa módon másoknak viszont a „sótlanság" nem tetszik: „egy kis humor nem ártana a sok tömény számítástechnika közepette" — hangzik az ellentétes vélemény. Van, aki „minden bájtra tömény információt" igényelne, van viszont, aki azért hálás, mert — kevésbé avatott szakember lévén — az antropomorf megfogalmazás közelebb hozza őt egy-egy témakör megismeréséhez.

Szó érte a ház elejét címadás ügyben: többen túlzásnak tartják a zsurnaliszta blickfangot és inkább a témát első ránézésre egyértelműen eláruló címeket szeretnének. Címadási stratégiánkon — amelynek lényege, hogy többnyire a felcím hordozza a szakterületre vonatkozó utalást, a főcím pedig áttételesen vagy játékosan kiemeli a cikk valamelyik gondolatát, mondanivalóját — nem szeretnénk változtatni. Felfogásunk szerint ugyanis az ismeretanyag átadását, a lényeg megragadását egy jól eltalált asszociáció hatékonyabban szolgálja, mint a száraz, szolgaian tematikus cím.

„A rovatok száma túl nagy, fölöslegesek a semmitmondó rovatnevek (Fogódzó, Paletta stb.)" — írja egyik olvasónk, aki „A hónap témája"-szerű feldolgozást viszont egy egész lapszám megtöltésére is alkalmasnak ítélné. Úgy érezzük, a rovatszerű tagolás inkább erénye lapunknak, és inkább javítja az információk fogyaszthatóságát. A rovatcímadás lényegében hasonló elveken alapul, mint az egyéb címeké, ha nem is mindig sikerül elég frappánsan kifejezni a rovat szerkesztése mögötti szándékot. Célszámokat azért nem tervezünk, mert „A hónap témája" eleve betölt ilyen szerepet (néhány olvasónk még annak terjedelmét is sokallja), továbbá azokra is gondolni kell, akik a többi témakört és rovatot szeretnék minden számban látni. Ez a magyarázata annak is, hogy átfogó összeállításunkban nem szívesen dolgozunk fel szűkebb szakmai témákat, bár azok jelentőségével is tisztában vagyunk.

Mint lapunk figyelmes olvasói észrevehették, nem ragaszkodunk ahhoz, hogy egy-egy rovatot mindenáron életben tartsunk. Többen hiányolták például az olvasói leveleket. Ennek legfőbb oka az, hogy általában sikerült teljes értékű cikké érlelni és más rovatban elhelyezni a spontán módon beérkezett észrevételeket, témafelvetéseket, ezért a Visszacsatolás rovatnak kevesebb anyag jutott.

A kérdőíven szereplő megjegyzésekben többen felvetik, hogy van-e lehetőség amatőrként készített, de jó és hasznos programok közlésére. Boldogan fogadunk, gondozunk és közkinccsé teszünk minden olyan alkotást, cikket, programot, ötletet, amely lapunk olvasótáborának érdeklődésére tarthat számot. Tehát „amatőrök" se kíméljenek! Vállaljuk még a közvetítő szerepet is, ha a termék esélyesnek látszik a kereskedelmi forgalmazásra.

A témák feldolgozási módját illetően sok jogos kritika ért bennünket. Legtöbben a véget érni nem akaró sorozatokat kifogásolják. Ígérhetjük, hogy részben saját tapasztalataink, részben éppen e felmérés alapján a témák, rovatok egyre jobban közelítenek majd az olvasói igényekhez.

Bár néhány olvasónk hiányolja, mi egyre kevésbé látjuk indokoltnak, hogy kinyomtatott programlistákat rendszeresen közöljünk. Arra ott van a lemezmelléklet! A Mikroszámítógép Magazinnak az 1990. júniusi számtól Alaplappá történt átalakításával lényegében búcsút intettünk a Sinclair-, Commodore-, Atari-, TVC-, Enterprise- stb. világnak. Tudjuk — amint ezt sok olvasónk meg is jegyzi —, hogy máig is betöltetlen az űr, amelyet ez a „szakítás" hagyott maga után, de a lap fennmaradását egyedül a PC-profilra való átállás tette lehetővé.

A másik alapvető kérdés, hogy milyen megközelítésben foglalkozzunk a számítástechnikával. Mi elsődlegesen a szoftvert tekintjük fő „vadászterületünknek". A hardverrel továbbra is inkább csak a szoftver alkalmazásából kiindulva és ahhoz kapcsolódva szeretnénk foglalkozni, de természetesen elfogadjuk azokat az észrevételeket, amelyekben olvasóink több hardver vonatkozású anyagot, sőt hardvertanfolyamot is szeretnének látni az Alaplapban. Ennek első nyomaival már mostani számunkban is találkozhatnak.

A közvéleménykutatás első áttekintése igazolta feltevésünket, hogy olvasótáborunk szakmai szintjét tekintve nagyon heterogén és széles skálán mozog. Ezért továbbra is vállaljuk annak minden nehézségét, hogy az Alaplapban a kezdőktől a profikig mindenki találja meg saját hasznos olvasnivalóját. Ha egy téma a kezdők számára jól van feldolgozva, abból egy tapasztalt szakember is okulhat. Arról se feledkezzünk meg, hogy miközben valaki a számítástechnika valamelyik szűk területén szakértőnek számít, más témákban sokszor még csak az alapok elsajátításánál tart. És fordítva is: csak az írás minőségén múlik, hogy a „bájtfülűeknek" szánt cikk érthető és élvezhető lehessen az avatatlanoknak. Nem szégyellünk tehát egyes rovatokban — a kezdő felhasználókat támogatandó — egészen triviális szintekig „lemerészkedni".

Azzal, hogy olvasóink ilyen szép számban töltötték ki kérdőívünket és válaszoltak az „Egyéb javaslatai, észrevételei" pontra is, a szerkesztési munkában jól hasznosítható, értékes információkhoz juttattak bennünket. A vélemények sokfélesége olvasmányként is nagyon érdekes, mindenki számára tanulságos lehet, ezért azokkal később számaink Visszacsatolás rovatában rendszeresen találkozhatnak majd.

Varga János

A minőség megéri az árát!

Amikor elkezdtük, nemcsak mi, hanem a számítógép-vásárlók többsége is kezdő volt. Valamennyien elkövettük azt a hibát, hogy elhittük: érdemes a legolcsóbb, még éppen működő gépekkel foglalkozni. Az évek során azonban rengeteg tapasztalatot szereztünk, és ma már tudjuk: a számítástechnikában sincsenek csodák. A minőségnek és a megbízhatóságnak ára van, s ha ezen spórolunk, az mindkettő rovására megy.



A fejlett piacgazdaságú Nyugaton azt tartják: a nagyon olcsó, bóvli termék a szegények adója. Aki mégis megveszi, rövidesen tapasztalja: rossz üzletet kötött.

A mindenáron való árcsökkentés ugyanis odavezet, hogy egyre gyengébb lesz a minőség, míg végül elérkezünk arra a pontra, amikor ez már nem éri meg a még oly olcsó árat sem. Mi nem akarjuk, hogy ügyfeleink később azt mondják: átvertük őket, egy vackot sóztunk rájuk, ezért inkább nem is forgalmazunk a legalsó árkategóriába tartozó számítógépeket és nyomtatókat. Úgy gondoljuk: egy átlagosan jó minőségre minden vevőnek joga van. Mi ezért nem tekintjük jelszónak, hogy nálunk **a minőség mindig megéri az árát.** Ez több annál, **ez a mi üzleti filozófiánk.**

De ez az ár nem feltétlenül magas — sőt! Először is az **IR** tudja, hogy sokféle igény, feladat és természetesen sokféle pénztárca létezik. Ezért többféle áron, többféle minőséget kínál, az átlagos felhasználónak megfelelő, olcsó „standard" típustól a profiknak szánt, előtesztelt alkatrészekből szerelt, 72 órán át járatott „extra" kategóriáig, amelyhez kétéves garanciát adunk.

De nemcsak ezért lehetséges, hogy az **IR Szervizben** mindenki megtalálja az igényeinek és anyagi lehetőségeinek megfelelő számítógépet. Segíti ezt az is, hogy az **IR** ma már akkora forgalmat bonyolít le, hogy szállítói megadják neki a legjobb vevőnek járó árkedvezményeket. Az **IR** pedig ezeket **a kedvezményeket maradéktalanul átadja** a hazai számítógép-felhasználóknak.

Így lehetséges, hogy egy kiváló minőségű 286-os AT, 40 Mbájtos merevlemezzel, 1 Mbájt RAM-mal, floppyval és nagyfelbontású, színes VGA monitorral **10%-kal olcsóbb az átlag hazai árnál.** Hasonló a helyzet a hordozható számítógépeknél is. De ugyanilyen jók az árai az egyedi igények alapján összeállított konfigurációknak és hálózatoknak is.

És ez még nem minden. **A nyári hónapokra időszakos számítógépvásárt hirdet az IR Szerviz.** Ennek során — konfigurációtól függően — akár **50 000 forintos árkedvezményt** is kapnak a vásárlók. Az oktatási intézmények pedig további nagyon jelentős, **25%-os kedvezményt** élveznek.

Természetesen a választás joga az Öné.
Keressen fel bennünket Budapesten,



IR Szerviz
VII. kerület, Kis Diófa utca 6.
Tel.: 121-3230, 141-0880

írjon levelet, vagy egyszerűen hívjon telefonon, a 121-3230 vagy 141-0880-as számokon, és mi segítünk, hogy kiválassza az Önnek megfelelő számítógépet. **Bármit választ, nyugodt lehet: olyan minőséget kap, amely biztosan megéri az árát!**

INFORMÁCIÓKÉRÉS: 12 ▲

Múlt, jelen, jövő

E havi összeállításunk anyagai leginkább gazdaságunk állapotának időtényezőjével függnek össze.
Három termék, három „időzóna".
Az egyik az örökségként kapott gyatra telefonhálózat feltételei közepette is képes a megbízható adatkommunikációra.
A másik a ma legdinamikusabb kisvállalkozói szféra számára ad a kiadványkészítéshez egy viszonylag olcsó, de igen jól használható eszközt.
A harmadik, a világpremierrel csaknem egy időben itthon is megjelent szuper gépcsalád — egy darabka jövő.

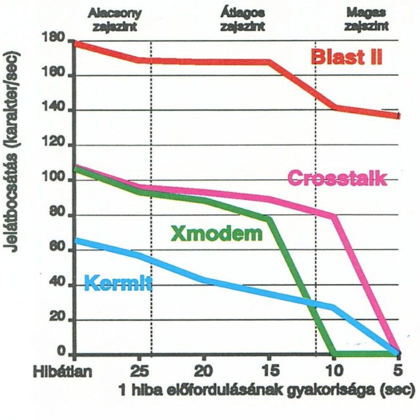
BLAST a magyar vonalakra

Még a tavaszi Ifabón mutatták be az amerikai fejlesztésű BLAST programcsomagot, amely a különböző típusú számítógépek adatainak összekapcsolását teszi lehetővé a gyenge minőségű magyar telefonvonalakon is.

A BLAST-tal állományokat hozhatunk át közeli vagy távoli számítógépekről, illetve állományokat vihetünk át közeli vagy távoli számítógépekre. Az átvitel a gyors és hibamentes BLAST eljárással történik. A BLAST átküldés előtt különböző stratégiák szerint tömöríti az adatokat, vagyis fizikailag kevesebb bájtot kell átküldeni, így kevesebb is sérülhet meg a vonalon. A vonal névleges áteresztőképességét akár két-háromszorosával is meghaladhatja a fájlban levő adatok mennyisége. A BLAST-ra jellemző az is, hogy a kapott adatokat nem írja fel azonnal a lemezre, hanem összegyűjti a hibátlan adatcsomagokat, és csak a hibások ismétlését kéri. Ez nagy előnye például a közismert Kermittel szemben, ahol a telefonvonal megszakadásakor a gyűjtés elölről kezdődik.

A BLAST duplex módon kommunikál a vonalon, vagyis egyidejűleg mindkét partner küldhet és kaphat információkat.

Adatátviteli összehasonlító teszt növekvő zajszint mellett



Lehetőség van arra is, hogy parametrizáljuk a vonalon az adatcsomagok méretét: kevésbé zajos vonalon nagy adatcsomagok, zajos vonalon kis adatcsomagok küldhetők.

A BLAST bármikor áttérhet terminálemulációs módba, így közvetlenül dolgozhatunk a számítógépen úgy, mintha az egy másik számítógép terminálja lenne. Terminálemulációnál lehetővé válik az is, hogy egy távoli gép esetleg a miénktől eltérő operációs rendszerével dolgozzunk.

A BLAST elterjedését megkönnyítheti, hogy átkapcsolható más adatátviteli szabványokra is (Kermit, Xmodem, Ymodem). A BLAST-nak saját védelmi rendszere van, vagyis vissza nem fejthető jelszóval megakadályozza az illetéktelen bejelentkezéseket. Több mint 30 operációs rendszerre egységesen kidolgozták, így az operációs rendszer váltásakor is jól használhatjuk.

A BLAST-ot kizárólagos joggal forgalmazó Comfort Kft.-nél megtudtuk, hogy szeretnék, ha ez a termék standard átviteli eszközzé válna. Akkor ugyanis mindenki mindenkivel tudna adatot cserélni, a felhasználóknak módjuk lenne egységes, valamennyiük által jól ismert szoftvertermékek segítségével gépeik között kapcsolatot teremteni. Nem kellene egyedi, többnyire kényelmetlen és munkaigényes eszközökkel kínlódniuk. A BLAST hazai elterjedését segítheti az is, hogy rendkívül olcsó: PC-s változatát 24 000 forintért, VAX-os változatát pedig 95 000 forintért árusítják. A programcsomag kezelése könnyen elsajátítható, a Comfort Kft. a szoftver mellé magyar nyelvű felhasználói kézikönyvet is ad.

Scanner a saját kiadványkészítéshez

A nyáron mutatta be az Intercomp Kft. azt az A/4-es scannert, amely kedvező árával mindazon kisvállalkozóknak elérhető, akik maguk szeretnék elkészíteni kiadványaikat (étlapokat, árlistákat, emblémákat, termékismertetőket stb.).

A Tajvanról származó scanner mind fekete-fehér (ScanPlus, kb. 70 000 Ft), mind pedig színes (ScanColorPlus, kb. 120 000 Ft) változatban forgalomba kerül. A scannerrel együtt jár az illesztőkártya és a Picture Publisher szoftver is, amellyel az elektronikus képet szerkeszthetjük.



A scannerhez tartozó szoftver Windows alatt fut. A program .TIF fájlokat állít elő, s ezeket már az ugyancsak Windows alatt futó szövegszerkesztők is tudják értelmezni (Word for Windows). A szoftver ismeri az összes editálási funkciót, 256 színárnyalat állítható elő segítségével. Az ismert kicsinyítési, nagyítási, retusálási, alakkivágási, forgatási, nyújtási funkciók mellett átmehetünk „negatívba", vagy akár a képpontok szintjén szerkeszthetünk. A képek editálásakor minden tevékenység végrehajtása után eldöntheti a felhasználó, hogy elfogadja-e a módosítást vagy sem. Javításkor így tehát semmilyen „próbálkozásunk" sem veszhet el.

A fekete-fehér scannerrel 64 szürke árnyalatot, valamint a képek kontrasztját állíthatjuk be. Gyakran van szükség a scanner úgynevezett prescan („próbafelvétel") funkciójának használatára. Ezzel a funkcióval kisebb felbontásban megjeleníthetjük a teljes A/4-es lapot, és ezen a vázlatképen kijelölhetjük azt a részt, amelyikre tulajdonképpen szükségünk van. A beolvasott és megszerkesztett képet azonnal ki is nyomtathatjuk mátrix- vagy lézerprinterre, meghatározva a kinyomtatandó kép helyét a papíron.

Amikor nem nyomdai technológia részeként van szükség a scannerre, hanem nyomtatványok házi előállításához és másológéppel való sokszorosításához készítünk anyagot, remekül lehet használni ezt a scannert és a hozzá tartozó szoftvert.

SzuperNOVA?

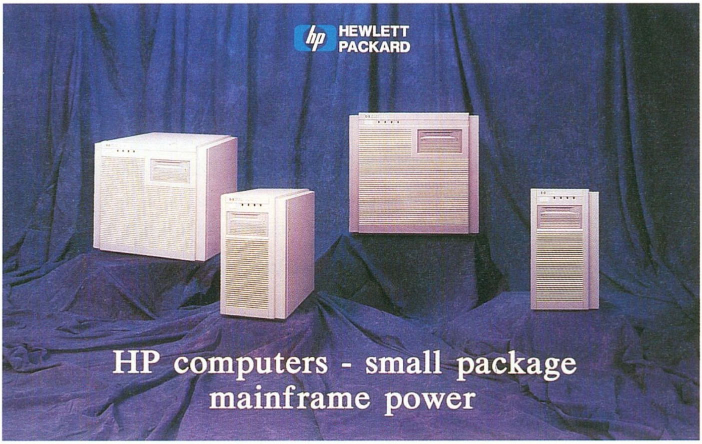
A Hewlett-Packard új, NOVA termékcsaládja az amerikai premier után röpke két héttel megjelent a hazai piacon is, a Hewlett-Packard & Controll Kft. jóvoltából. A sajtótájékoztatón a HP & C vezetői megígérték, hogy a szaksajtó képviselői szeptemberben már fizikai valóságában is tesztelhetik az új gépcsaládot.

Az új termékcsalád a RISC-technológián alapul: a rendszerek teljesítményét a precíziós architektúrájú RISC-chipek biztosítják. A rendszerek a HP 3000 és HP 9000 számítógépcsalád kiegészítő tagjai, amelyek teljesítményben hatszorosan (!) felülmúlják a hasonló árkategóriába tartozó IBM és DEC termékeket. Ezzel a gépcsaláddal azokat a nagyvállalatokat és intézményeket — elsősorban banki, államigazgatási szférában — akarják megcélozni, amelyeknél nagyon fontos a számítógépes rendszerek kompatibilitása, kezdve az asztali készülékektől egészen az adatfeldolgozó központokig.

A HP & C Kft. szerint Magyarországon is lesz igény az ilyen nagy teljesítményű mini-, illetve szupermini számítógépekre, ugyanis az egyre bonyolultabb alkalmazások nagy tudású berendezések használatát teszik szükségessé. A NOVA gépeknél — amelyek akkorák, mint egy élére állított PC — nincs szükség különleges tápegységre, hűtésre. A CPU, a merevlemez, a mentési lehetőség és az operációs rendszer eleve bennük van, sőt a berendezéshez hálózati szoftver is tartozik. Tehát a gép „magától talpra áll", a felhasználónak csak be kell kapcsolnia, és már dolgozhat is vele. A kis gépen kifejlesztett alkalmazásokat pedig változtatás nélkül lehet futtatni a nagy gépen is.

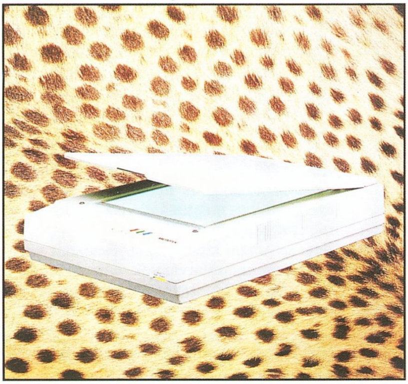
A HP & C „agresszív" marketing-politikával igyekszik megszerezni a magyar számítástechnikai piac minél nagyobb szeletét. Azoktól a vevőktől, akik áttérnek a HP számítógépeire, visszavásárolják addigi bármilyen gyártmányú, működő számítógépüket (legyen az Bull, IBM, Digital, ICL...), és még 12 000 dolláros hitelt is adnak az átálláshoz. Kíváncsian várjuk, hogy ilyen paraméterekkel és ilyen üzletpolitikával milyen mértékben — és mikor? — terjednek majd el ezek a gépcsodák.

Sziebig Andrea



A LETAPOGATHATÓ VALÓSÁG

SCANMAKER 600 Z/ZS



MICROTEK

Scanner · Software · Support

MICROTEK SCANNER

Felbontás: 600 DPI.

Képméret: max. 21,6 x 35,6 cm.

Letapogatás: háromszori átfutással, külön a piros, a zöld és a kék színhez.

Csatlakozás: IBM PC, PS/2 és Apple Macintosh számítógépekhez.

A Microtek magyarországi disztribútora:

Mikropo
Számítástechnikai
Kisszövetkezet
1369 Bp.
VI., Nagymező u. 51.
Telefon: 112-7830,
Fax: 112-4431

MIKROPO

INFORMÁCIÓKÉRÉS: 35 ▲

RENDSZERVÁLTÁS

A SZÁMÍTÁSTECHNIKÁBAN!®

Eddig PC-k (XT-től — 486-ig), alkatrészek, perifériák, nyomtatók és egyéb kiegészítők forgalmazásával foglalkoztunk.

Most mindezeket hálózatba kötve, telepítve, bevizsgálva TPA és jogtiszta DEC rendszerekbe is integráljuk.

3M TERMÉKEK

Floppylemezek, streamer-kazetták

SZÁLLÍTÁS RAKTÁRRÓL

AZ ÉRDEKLŐDŐKET VÁRJUK MEGNYÍLÓ BEMUTATÓTERMÜNKBEN!

Kérje részletes árlistánkat!

TALÁLKOZZUNK A COMPFAIR '91-EN AZ A/308-AS STANDON!

MACRODA

MACRODA KERESKEDELMI KFT.

1123 Budapest, Alkotás u. 21. Telefon: 156-4802

INFORMÁCIÓKÉRÉS: 11 ▲

Ha a megbízhatóság a döntő...



VIGYÁZAT! Jól bevezetett és hírnévnek örvendő márkanevünkkel kétes minőségű, hasonló hangzású nevek élnek vissza!

Forgalmazó:

INTERAG

INTERAG INFORMATIKA
Budapest 1136 Pannónia u. 11.
Tel./fax.: 132-9375 Molnár Péter, Sugár Mihály

People Committed To InfoTech

MiTAC

INFORMÁCIÓKÉRÉS: 08 ▲


NTT—2000
Trade and Service, Ltd.
A RANK XEROX
hivatalos dealere
RANK XEROX
fénymásolók, faxok,
lézerprinterek,
írógépek forgalmazása
RANK XEROX
berendezések
színvonalas szervizellátása
RANK XEROX
kellékanyagok biztosítása
Cím: 1431 Budapest
VIII., Mária u. 20.
Telefon: 1340-900, 1340-393
Fax: 1340-568
XEROX 7009
INFORMÁCIÓKÉRÉS: 10 ▲